Maestro Command Reference Manual

Version 7.0, January 2005

For inquiries about Maestro ™ contact:

Schrödinger
1500 SW First Avenue, Suite 1180
Portland, OR 97201–5881
503–299–1150
503–299–4532 fax
Email: help@schrodinger.com

Copyright © 2005, Schrödinger, LLC. All rights reserved.

Schrödinger, FirstDiscovery, Glide, Impact, Jaguar, Liaison, LigPrep, Maestro, Phase, Prime, QikProp, QikFit, QikSim, QSite, and Strike are trademarks of Schrödinger, LLC. MacroModel is a registered trademark of Schrödinger, LLC.

To the maximum extent permitted by applicable law, this publication is provided "as is" without warranty of any kind. This publication may contain trademarks of other companies.

Please note that any third party programs ("Third Party Programs") or third party Web sites ("Linked Sites") referred to in this document may be subject to third party license agreements and fees. Schrödinger, LLC and its affiliates have no responsibility or liability, directly or indirectly, for the Third Party Programs or for the Linked Sites or for any damage or loss alleged to be caused by or in connection with use of or reliance thereon. Any warranties that we make regarding our own products and services do not apply to the Third Party Programs or Linked Sites, or to the interaction between, or interoperability of, our products and services and the Third Party Programs. Referrals and links to Third Party Programs and Linked Sites do not constitute an endorsement of such Third Party Programs or Linked Sites.

Maestro Version 70110
January 2005

1 The Maestro Command Language

This manual contains a listing of all supported Maestro commands, their functions, and their usage options. Use Maestro commands by entering them in the Command Input Area of the Main Application Window.

The Maestro command language syntax is:

keyword [*options*] [operand]

For example:

entryimport *wsreplace*=true *all*=true *format*=maestro ligands.mae

In the example given, "entryimport" is the keyword. As this example illustrates, the first item entered must be a keyword, and this keyword must match a known keyword completely, i.e., to all characters. However, Maestro does have an alias facility, which allows users to map keywords to shorter names, for example, "entryimport" to simply "import". See the "alias" command for details.

Command options are used to set values that generally correspond to states within the program. As the name "options" suggests, inclusion of these items in a Maestro command is not mandatory. However, any number of option *name*=⟨value⟩ pairs can be specified. Options can be of type: Boolean (yes/no, y/n, t/f, and true/false are valid values), string, integer, or real.

Once an option is set, its value will persist for all subsequent commands until it is specified again. A command consisting only of a keyword and options will set the states corresponding to those option values, but will not perform any other action. For example, the command:

entryimport *format*=maestro *all*=true

will not import structures from any file. It will only set the format for subsequent file reading and specify that all structures contained in the subsequently specified file be imported. If the following command is then issued:

entryimport ligands.mae

all structures from the ligands.mae Maestro file will be imported.

The operand is the remaining element of a Maestro command. The type of operand required is dependent on the specific command. For "entryimport", the operand must be a file name. For other commands, it may be one or more atom numbers, or a string describing arbitrary atom sets, specified using Maestro's Atom Specification Language (ASL). For more information about ASL, see Chapter 3 [ASL], page 5.

2 Conventions Used in this Manual

This manual uses a number of typographical conventions to describe the command language.

The names of keywords are printed in boldface:

entryimport

The names of options are printed in italic, and literal choices for that option are in normal roman:

wsreplace=true

Values which the user is to replace with appropriate values are represented in angle brackets:

entryimport ⟨file_name⟩

Where there is a choice of a number of values, these are shown separated by vertical bars:

entryimport *format*=maestro|mmod|sd|pdb|mol2

If a value is optional then it is enclosed in square brackets:

print [⟨file_name⟩]

The following conventions are used to describe acceptable values for options:

⟨n⟩	means an integer
⟨x⟩	means a real number
yes|no	means a boolean value (could also be true|false, y|n, on|off)
⟨text⟩	represents a string value. If this contains spaces, it must be enclosed in double quotation marks

The following are some conventions used for operands:

⟨atom_number⟩	represents an atom number
⟨atom1⟩	represents the first atom number in the operations, ⟨atom2⟩ the second, and so on
⟨ASL⟩	represents a valid string in the Atom Specification Language
⟨ESL⟩	represents a valid string in the Entry Specification Language

3 The Atom Specification Language

This chapter documents the syntax of Maestro's Atom Specification Language (ASL).

3.1 Why an Atom Specification Language?

- To provide a flexible way to define sets of atoms in complex macromolecular systems
- To serve as the basis of a "Sets" facility in Maestro
- To allow atom specification from textual input which some users find faster then picking atoms from the main structure window.

3.2 The ASL Hierarchy

`entry > molecule > chain > residue > atom`

There are five classes which make up the atom specification language. Each is listed below.

In this section the minimum acceptable abbreviation is shown outside the square brackets. So for example `m[olecule]` means that the minimal acceptable abbreviation for `molecule` is `m`.

`e[ntry]` This is the top level class in the language. An entry is all atoms in the workspace associated with a single entry in the currently open project.

`m[olecule]`
The term molecule is used in the normal chemical sense meaning all atoms which are connected by a single covalent path.

`c[hain]` This corresponds to a chain as specified in the PDB file format. Note that this chain may be a subset of a molecule, e.g., when chains a linked by disulphide linkages.

`r[esidue]`
An arbitrary collection of one or more covalently bound atoms within a molecule, such as the monomer units in a polymer.

`a[tom]` A single atom.

Each class is optional. If absent all entities of that type are matched.

3.3 Atom Specification

A complete specification is a class name and some property specified by a property name and property list. The syntax is:

class.property ⟨ propertylist ⟩

Items in a property list may be separated by comma, whitespace or both. Ranges (lower-upper) may be used where appropriate. Unterminated ranges are taken to include all available numbers. For example, if there are four molecules in the system then the specifications:

- `mol. 2, 3, 4`
- `mol. >=2`
- `mol. 2-4`
- `mol. >1`

are equivalent.

In a similar manner,

- `mol. 1, 2, 3`
- `mol. <=3`
- `mol. 1-3`
- `mol. <4`

are equivalent.

All names of properties and characters in property lists are treated in a case insensitive manner.

Wildcards are supported for atom and set names. A '`*`' will match zero or more characters and a '`?`' will match any single character. You can include comments in a specification by placing a '`#`' character before the text you wish to hide.

In this section the minimum acceptable abbreviation is shown outside the square brackets. So, for example, `a[tom].pt[ype]` means that the minimal acceptable abbreviation for `atom.ptype` is `a.pt`.

`e[ntry].`

`[name]` Because entry names are the only entry properties specifiable using ASL expressions, the word `name` can be completely omitted, i.e., `entry.name ename` and `entry. ename` are equivalent. Note, however, that the '`.`' is still required even if the property name is not included. A valid property list for the 'entry name' property is a list of entry names. Wildcard characters are permitted. For example:

```
entry. e1
entry.name recep, lig*
```

`m[olecule].`

`[number]` Because numbers are the only molecular properties specifiable using ASL expressions, the word `number` can be completely omitted, i.e., `mol.number 1` and `mol. 1` are equivalent. Note, however, that the '`.`' is still required even if the property name is not included. A valid property list for the 'molecule number' property is a set of numbers or a range. For example:

```
mol. 1-4
mol. 1,2,3,4
```

`m[odulo]` a property which can be used to select every *n*th molecule. For example: `mol.mod 10 1` will select molecule 1, 11, 21, etc.

`e[ntrynum]` a property which can be used to select molecules based on their entry-relative numbering. For example: `mol.entrynum 1` will select the first molecule in each entry.

`a[toms]` a property which can be used to select molecules based on the number of atoms they contain. For example: `mol.atoms 200` will select molecules that contain exactly 300 atoms. Other examples are:

```
mol.atoms 200-300
mol.a > 200
```

`w[eight]` a property which can be used to select molecules based on their molecular weight. For example: `mol.weight 200.12` will select the atoms that have a molecular weight of exactly 200.12. Other examples are:

```
mol.weight <=300.0
mol.weight > 200.0
```

`c[hain]` This class designation allows you to specify atoms using chain attributes. Combine with the 'name' property.

`[name]` Because names are the only chain properties that are specifiable by ASL, the word `name` can be omitted. For example, the specifications `chain.name A` and `chain. A` are equivalent. Note, however, that the '`.`' is required even if the property name is not included. A valid property list for the 'chain name' property is a single character representing a PDB

chain name. Some examples of equivalent acceptable 'chain name' expressions are:

```
chain.name A
chain. A
c. A
```

`r[esidue]`

This ASL class designation allows you to specify atoms based on residue properties. Combine with one of the following property specifications.

`[name]` or `[number]`

Either residue names or numbers (but not both) can be used in property lists which do not specify a property name For example, the following are valid ASL expressions:

```
res. ala val leu
res. 1 2 3
```

and will return atoms which are either in alanine, valine, or leucine residues, or atoms in residues 1, 2 or 3, respectively. Residue numbers can be specified with a range, e.g., `residue. 1-4` and may include negative values or zero.

`pt[ype]` The three-letter PDB code for the residue. This is the default for non-numeric characters in the property list, so the expression `res. arg` and `res.ptype arg` are equivalent. A valid property list for 'ptype' is comprised of three-character tokens. For example:

```
res.ptype gly,val,ala
res. gly val ala
```

`m[type]` The one letter residue codes as used in Maestro. A valid property list for 'mtype' is comprised of one-character tokens. For example:

```
res.mtype g,v,a
res.m g,v,a
```

`po[larity]`

The polarity of the residue. The property list must consist only of the following descriptor types:

`h[ydrophobic]`
returns atoms in hydrophobic residues

`pol[ar]` returns atoms in polar residues

`pos[itive]`
: returns atoms in residues with positive formal charges

`n[egative]`
: returns atoms in residues with negative formal charges

For example,

```
residue.polarity hydrophobic
residue.pol pos,neg
res.pol h pos neg
```

`sec[onadary_structure]`
: The secondary structure of the residue. The property list must consist only of the following descriptor types:

`h[helix]` returns atoms in helical regions

`s[trand]` returns atoms in strand regions

`l[oop]` returns atoms in loop regions

For example,

```
residue.sec helix
residue.sec hel, str
res.sec l, s
```

`pos[ition]`
: The fractional position of the residue. The property list must include two real numbers representing a fractional range of residue numbers. For example, if there are 100 residues numbered from 1-100, the specification:

```
residue.pos 0.0 0.1
```

will return residues 1 to 10.

`i[nscode]`
: The insertion code of the residue. A property list should include one-character tokens representing insertion codes. For example:

```
residue.inscode a
```

will get all residues with insertion code 'a', while:

```
res. 25 and res.inscode b
```

will get residue 25b.

`a[tom]` Atom ptypes and numbers may be mixed in property lists where no explicit property is specified. For example, the following is valid:

```
atom. 1,2,3,CA
```

and returns atoms 1,2 and 3 and any alpha carbons.

`pt[ype]` The PDB atom names. A valid property list for 'ptype' consists of acceptable PDB names. This is the default property for non-numeric components of property lists and, as such, the word 'ptype' may be omitted. Note that the '`.`' is required. The following specifications are equivalent and return the "backbone" atoms in a structure:

```
atom.ptype N,CA,C,O
atom. N,CA,C,O
a. n,ca,c,o
```

Note: see below for a discussion of how PDB atom names are specified and matched.

Wildcards as described above can be applied to ptypes.

`na[me]` The atom names. The property list must contain valid atom names. A valid atom name could be a string of any length which:

1. must have at least one non-digit character
2. must not contain any control characters (ASCII value < 0x20)
3. must not contain spaces or equal signs, unless the name is quoted

Wildcards as described above can be applied to atom names. Examples:

`atom.name the_36th_carbon`
`atom.na C15, O:66, H-77`
`atom.na C*` (returns atoms with name starting with C)
`atom.nam ??O*` (returns atoms whose name's 3rd character is '`O`')

`n[umber]` The atom numbering. The property list must be a list or a range of numbers. This is the default property for numeric components of property lists. The following expressions are equivalent and return the atoms numbered 1, 2, 3, and 4.

```
atom.num 1,2,3,4
a. 1 2 3 4
atom. 1-4
```

`mo[lnum]` The atom numbering in the "by molecule" scheme. The property list must be a list or a range of numbers. For example, the expression:

```
atom.molnum 1
```

returns the first atom in each molecule, while the specification:

```
atom.molnum 1-10
```

returns the first 10 atoms in each molecule.

`en[trynum]`
The atom numbering in the "by entry" scheme. The property list must be a list or a range of numbers. For example, the expression:

```
atom.entrynum 1
```

returns the first atom in each entry, while the specification:

```
atom.entrynum 1-10
```

returns the first 10 atoms in each entry.

`m[type]` The Maestro atom type. A valid property list for the 'mtype' property consists of Maestro atom types. The following expression is valid and specifies sp2 carbons and oxygens.

```
atom.mtype C2,O2
```

`e[lement]`
The element symbol for the atom. A valid property list for the element is a list of standard periodic table symbols. To define all carbons and oxygens:

```
atom.ele C,O
```

`att[achments]`
The number of bonds the atom has to it. The property list must be a number in the range 0-6, but greater than '>', less than '<', and equals '=' signs may also be used. The expression:

```
atom.att 1
```

returns all terminal atoms. The specification:

```
atom.att <=2
```

returns all terminal atoms and other atoms with 2 or fewer bonds. The specification:

```
atom.att 0
```

returns all isolated atoms.

`ato[micnumber]`
: The atomic number. The property list must contain numbers only. Ranges of integers and greater than '>', less than '>', and equals '=' signs may also be used. The expression:

```
atom.atomicnum 1
```

returns all hydrogen atoms. The specification:

```
atom.ato 1-6
```

returns all atoms in the range H to C.

`c[harge]`
: The partial charge on the atom. A valid property list contains a value or range of floating point values. The expression:

```
atom.charge 0.400
```

returns atoms with partial charges of 0.400. The expression:

```
atom.charge -0.6--0.4
```

returns atoms with partial charges -0.6 to -0.4,

```
atom.charge <0.0
```

returns atoms with negative partial charges, and

```
atom.charge >=0.5
```

returns atoms with charges of 0.5 or greater.

`f[ormalcharge]`
: The formal charge on the atom. A valid property list contains a value or range of integer values. The expression:

```
atom.formal 0
```

returns atoms with formal charges of 0. The expression:

```
atom.formal -2 -- 1
```

returns atoms with formal charges -2 to -1,

```
atom.formal <0
```

returns atoms with negative formal charges, and

```
atom.formal >=1
```

returns atoms with formal charges of 1 or greater.

`d[isplayed]`
: Whether or not the atom is currently displayed in Maestro. No property list is used. For example, the expressions:

  `atom.displayed` returns the set of all displayed atoms
  `not atom.disp` returns all atoms not currently displayed
  `sidechain and fillres atom.disp` returns the side chains of residues where at least one atom is displayed.

3.4 Operators

A number of operators are supported:

3.4.1 The Boolean `and` Operator

Boolean `AND` (set intersection). The syntax for this operation is:

```
spec1 and spec2
```

where `spec1` and `spec2` are valid atom specifications. This operation will return the set of atms which meets the specifications `spec1` and `spec2`. For example, the expression:

```
mol. 1 and atom. CA
```

will return the set of all the alpha carbons of molecule 1. The specification:

```
res.num 1-100 and res. ala
```

returns all alanines in residues with numbers in the range 1-100.

3.4.2 The Boolean `or` Operator

The Boolean `OR` operator. The syntax for this operations is:

```
spec1 or spec2
```

where `spec1` and `spec2` are valid atom specifications. This returns the set of atoms which meet either specification `spec1` or `spec2`. For example, the expression:

```
mol. 1 or atom.ptype CA
```

returns the set of all atoms that are in molecule number 1, or are alpha carbons. The specification:

```
res.num 1-100 or res.ptype ala
```

returns all residues either with numbers in the range 1-100, or any alanines.

3.4.3 The Boolean `not` Operator

The Boolean `NOT` operator. The syntax for this operation is:

```
not spec1
```

where `spec1` is a valid atom specification. This returns the set of atoms that are not part of those defined by `spec1`. For example, the expression:

```
not atom. CA,C,N,O
```

will return a set containing all side chain atoms.

3.4.4 The `fillres` and `fillmol` Operators

Two special operations, `fillres` and `fillmol`, can be used to "fill out" the atoms defined by a atom specification to complete residue or molecule boundaries. For example:

```
fillres atom.num 1,100,40
```

will return all the atoms in residues of which atoms 1,100 and 40 are members. In a similar way:

```
fillmol atom.num 1,100,40
```

will return all the atoms in molecules of which atoms 1,100 and 40 are members.

3.4.5 The `within` and `beyond` Operators

The operators `within` and `beyond` can be used to define sets of atoms based on their distance from a set of atoms defined defined by an atom specification. The syntax for these operators is:

```
within distance spec1
```

which returns the atoms within, i.e, less than or equal to the distance in Angstroms of the set defined by spec1, and

```
beyond distance spec2
```

which returns the atoms which are further than the distance in Angstroms from the set defined by spec2.

For example, the expression:

```
within 5.0 mol. 1
```

returns the set of all atoms that are within 5 Å of molecule 1. The expression:

`beyond 5.0 mol. 2`

returns all atoms that are farther than 5 Å from molecule 2.

The combination of `fillres` and `within` or `beyond` is especially powerful.

`fillres within 5.0 mol. 1`

will produce a set containing the atoms of all complete residues that have atoms within 5 Å of molecule 1. Note that the `within` operator will also return the reference set of atoms:

`within 5.0 mol. 1`

returns the reference set of all atoms that are within 5 Å of molecule 1 and those that are part of molecule 1.

The `and` operator, when used with `within` and `beyond`, can be used to allow more specificity:

`mol. 2 and within 5.0 mol. 1`

returns the set of all atoms of molecule 2 that are within 5 Å of molecule 1.

3.4.6 The `withinbonds` Operator

A special type of `within`, one that finds all atoms within a certain number of bonds of the reference set:

`withinbonds` ⟨num_bonds⟩ ⟨spec⟩

For example:

`withinbonds 4 atom. 1`

will find all the atoms that are within four bonds of atom 1.

3.4.7 The `beyondbonds` Operator

A special type of `within`, one that finds all atoms beyond a certain number of bonds of the reference set:

`beyondbonds` ⟨num_bonds⟩ ⟨spec⟩

For example:

`beyondbonds 4 atom. 1`

will find all the atoms that are in the same molecule as atom 1 but beyond four bonds of atom 1.

3.5 Operator Priority

The order of priority of operators is (in decreasing order):

- `not/fillres/fillmol`
- `and/or`
- `within/beyond`

At equal levels of priority the expression will be evaluated left to right. Examples:

```
within 5.0 mol. 1 or mol. 2
```

returns the set of all atoms that are within 5.0 Å of either molecule 1 or molecule 2 (`or` has higher priority).

```
not atom.ptype CA,C,O,N and mol. 1
```

returns the side chain atoms of molecule 1.

```
atom.ptype CA or mol. 1 and not res.pol polar
```

returns all alpha carbons and atoms in hydrophobic residues of molecule 1. Parentheses can be used to override the order of evaluation:

```
not (atom.ptype CA,C,O,N or mol. 1)
```

produces all atoms either not in the backbone or not in molecule 1.

3.6 Implicit Operators

When no operator is specified, the following operations are assumed:

- Specifications within a property list have a Boolean `or` relationship. So the following specification:

  ```
  atom.ptype CA,CB
  ```

  matches any atom which has a PDB name of either CA or CB.
- When the classes are in order of decreasing priority in the class hierarchy, the `and` operator is assumed:

  ```
  mol. 1 chain. A atom.ptype CA
  ```

  is equivalent to:

  ```
  mol. 1 and chain. A and atom.ptype CA
  ```

  and both return the alpha carbons in chain A of molecule 1.

3.7 Creating New Sets from Existing Ones

Maestro supports a mechanism for defining and naming atom sets via its "Sets" panel. The names of existing sets may be used in expressions if they are prefixed with the word `set`. For example, if there are two sets defined as:

set S1 `mol. 1` (molecule number 1)
set S2 `atom.ptype C,O,N,CA` (all backbone atoms)

the following are then valid atom specifications:

- `set S1 and set S2`, all backbone atoms in molecule 1
- `set S1 or set S2`, all backbone atoms or atoms in molecule 1
- `within 5.0 set S1`, all atoms within 5.0 Å of molecule 1.

3.8 Special Specifications

- The specification `all` matches everything. You can use this in any Maestro command which expects an ASL operand in order to apply the command to all the atoms. For example to color all atoms green:

  ```
  coloratom color=green all
  ```

  Note that the syntax for this is just `all`, not `atom. all` or `molecule. all`.
- Anything enclosed in `/ /` is treated as a string in the linear substructure notation - see the BatchMin Reference Manual for a complete description of this. Some examples are:

  `/C3(-H1)(-H1)(-H1)/` specifies all methyl groups
  `/C2(=O2)-N2-H2/` specifies all amide groups
  `/C0-N0/` specifies all pairs bound C-N atoms, and
  `/O0-S0/` specifies any atom bound to a sulfur.

  Specifications made in this way can be treated in the same way as any other specification in the ASL, e.g. they can have operators applied to them.

3.9 Matching PDB Atom Names

Because PDB atom names are four characters wide we need to employ the following strategy in order to conveniently specify PDB atom names:

- Unquoted names which begin with a non-numeric character have a blank character inserted in front of them and are padded with blanks from the right to make up four characters before matching. Examples:
  - Property List: CA,C
    Actually Matched: `" CA "," C "`
  - Property List: CG1,CG2
    Actually Matched: `" CG1 "," CG2"`

- Unquoted names which begin with a number do not have an initial blank character inserted but are right to padded four characters. Example:
  - Property List: CA,1HB,2HB
    Actually Matched: `" CA ","1HB ","2HB "`
- Quoted names (double or single quotes are acceptable) are treated as literals but will be right padded with blanks to make up four characters. Examples:
  - Property List: CA,`"CA"`
    Actually Matched: `" CA ","CA "` (alpha carbons or calciums)
  - Property List: `" N A"`
    Actually Matched: `" N A"` (heme atoms)

3.10 Miscellaneous

3.10.1 Atoms not yet present in a structure

If a structure in the Workspace has only 100 atoms when the ASL definition: `atom.num 1,8,44,101,103` is issued, Maestro will simply match the atoms numbered 1,8 and 44. If additional atoms are subsequently added to the structure, the atoms bearing the numbers 101 and 103 will be added to the previously defined set.

3.10.2 Aliasing

Maestro allows you to define your own aliases, using either the Command Input Area or the Command Aliases panel. Maestro converts all aliases into their corresponding commands before performing operations involving the aliased commands. Users must ensure that aliases produce sensible results. Some aliases are supplied with the distribution. They are:

Operator: `and`
Aliases: intersection, INTERSECTION, &

Operator: `or`
Aliases: UNION, union, `|`

Operator: `not`
Aliases: !

Class Designator: `mol.`
Aliases: MOL, mol

Class Designator: `atom.`
Aliases: ATOM, atom

Class Designator: `res.`
Aliases: RES, res

Class Designator: `chain.`
Aliases: CHAIN, chain

ASL Definition: `atom. ca,c,n,h,o`
Aliases: BACKBONE, backbone

ASL Definition: `not (atom.pt ca,c,n,h,o)`
Aliases: SIDECHAIN, sidechain

ASL Definition: `"/H2-O3-H2/ or atom.mtype OW"`
Aliases: WATER, water

3.11 Useful Hints when using ASL with the Project Facility

Maestro 4.1 introduced its Project Facility. Entries can be included into and excluded from the Workspace. The order in which this is performed affects the molecule numbers. For example, if you have two entries in your Project Table called "A" and "B" and you include into an empty Workspace first A and then B, the molecule numbers will be 1 for A and 2 for B; however, if you first include B and then A, the molecule numbers will be 1 for B and 2 for A.

This means that an expression such as `mol. 1` will match different atoms in each of the above cases. In earlier versions of Maestro, it often made sense to use the `mol.` notation because there was no Project Facility. But with the Project Facility, Maestro is entry-centric and in most cases it makes more sense to use entry names.

For example, if you have an inhibitor and a receptor that are in different entries and wish to have a ribbon appear on only the receptor, use the entry name in the ASL expression, not the molecule number. This will ensure that when the receptor is included that it, and only it, will be used to generate the ribbon(s). A different inclusion order of entries in the Workspace will then result in the same matching atoms. So for ribbons with a receptor called `"receptor"` it is more useful to use `entry.name receptor` as the ASL definition.

3.12 ASL Examples

This section gives some examples of the use of the ASL in real-life situations. Note that while these examples all use lower-case, the ASL expressions themselves are not case sensitive.

i) Defining a set to refer to a ligand and/or receptor.

The exact command will depend on the nature of your system. If the ligand and the receptor are separate entries then it will suffice to use

```
set ligand entry.name <ligand_name>
```

where <ligand_name> is the name of the entry that contains the ligand. Similarly

```
set receptor entry.name <receptor_name>
```

for the receptor with entry called <receptor_name>.

In order to define sets that will work with multiple ligands it's also possible to define the ligand as everything that is not part of the receptor. A definition of:

```
set ligand not set receptor
```

will identify the ligand as anything that's not part of the receptor.

If the ligand and the receptor are part of the same entry then molecule numbers are the best way to define the ligand and the receptor. Assuming the receptor is molecule 1 and the ligand molecule 2:

```
set ligand mol.num 2
set receptor mol.num 1
```

Note however that the use of molecule numbers in set definitions should be avoided where possible as these depend on the order in which the project entries are included into the Workspace. If it is possible to use entry names, then these should be used.

The subsequent examples assume that sets for the receptor and the ligand have been defined using one of the methods defined above.

ii) The set of atoms within a given distance of the ligand.

One common task is to do something with the set of atoms within a given distance of the ligand. For example to only display those atoms or to include them in a substructure region for a MacroModel calculation. These examples will use the "displayonlyatom" command but the ASL which follows can be used with any other command that uses ASL.

To only display atoms within 5.0 Angstroms of the ligand:

```
displayonlyatom within 5.0 set ligand
```

A common variation is to display complete residues which have any of their atoms within a given distance of the ligand:

```
displayonlyatom fillres within 5.0 set ligand
```

It's also possible to restrict the expression so that it only applies to receptor atoms within a given distance of the ligand. Here the Boolean "and" operator is used to restrict the displayed atoms to the receptor only:

```
displayonlyatom set receptor and fillres within 5.0 set
ligand
```

Because this is a lengthy expression it's often convenient to make this into a set itself:

```
set active_site set receptor and fillres within 5.0 set
ligand
```

An equivalent form of this is:

```
set active_site (! set ligand) & fillres within 5.0 set
ligand
```

Note that "!" is a standard alias for "not" and similarly "&" for "and".

iii) Sidechain and backbone.

The ASL has standard aliases for the definition of sidechain and backbone atoms in proteins. For example to only display the atoms of the backbone:

```
displayonlyatom backbone
```

These aliases can be used with operators to build up more complicated expressions. For example to only display the sidechain of the receptor:

```
displayonlyatom sidechain and set receptor
```

To display only the sidechains of the atoms within 5.0 Angstroms of the ligand:

```
displayonlyatom sidechain and set receptor and fillres
within 5.0 set ligand
```

iv) Atoms of a given type.

There are a variety of ways to specify atoms of a given type. For example to specify all carbons, nitrogens and oxygens the following is used:

```
atom.ele C,N,O
```

To specify non-hydrogen atoms:

```
not atom.ele H
```

To specify the alpha carbons in a protein:

```
atom.ptype CA
```

To specify all sp2 carbons there are two choices. The first relies on knowing that the MacroModel atom type for such an atom is "C2" and using:

```
atom.mtype C2
```

The other (assuming no formally charged or radical carbons are present) uses the number of attachments to the atom:

```
atom.ele C and atom.att 3
```

To specify polar hydrogens:

```
atom.ele H and not /CO-HO/
```

or

```
atom.ele H and not atom.mtype H1
```

v) Water molecules.

The ASL has a standard alias "water". For example to delete all water molecules the Maestro command is:

```
delete atom water
```

vii) Restricting an operation to the atoms that are currently displayed in the workspace.

Often a user will be working with only a subset of the atoms in the workspace displayed. If an operation is to be performed only on the atoms that are displayed then the "atom.displayed" property can be used. For example to change the color to green of all the atoms currently displayed in the Workspace and to leave alone the undisplayed Workspace atoms:

```
coloratom color=green atom.disp
```

To only do it for the atoms that are displayed and in the receptor:

```
coloratom color=green atom.disp and receptor
```

viii) Specifying molecules.

All molecules with between 30 and 100 atoms:

```
mol.atoms 30-100
```

All molecules with over 100 atoms:

```
mol.atoms >100
```

All molecules with a molecular weight over 300:

```
mol.weight > 300.0
```

All molecules which contain a halogen:

```
fillmol atom.ele F,Cl,Br,I
```

ix) Specifying atoms based on a linear-substructure notation.

The ASL supports the use of a SMILES-like linear substructure notation to specify atoms with a particular bonding arrangement. The atoms are referred to by MacroModel atom types, but there are wildcard types that can be used to allow the expression to apply to any atoms of a given element type.

Some examples:

Any five-membered ring:

```
/00-00-00-00-00-1/
```

Aromatic six-membered carbon rings(C2 is sp2 carbon)

```
/C2-C2*C2-C2*C2-C2*1/
```

Amide groups:

```
/C2(*O2)-N2/
```

Methyl groups:

```
/C3(-H1)(-H1)(-H1)/
```

Water:

```
/H2-O3-H2/
```

Guanadinium group:

```
/N2(-H3)-C2(*N4(-H4)(-H4))-N2(-H3)(-H3)/
```

x) Using wildcard characters.

Most string-type property values can use wildcard characters. Some examples:

All PDB atom names beginning with C

```
atom.ptype C*
```

All forms of the histidine residue:

```
res.ptype HI*
```

All entries that begin with "lig":

```
entry.name lig*
```

xi) Addressing residues with negative residue numbers.

From version 6.1 of Maestro it is possible to use residue numbers that are negative or zero.

```
res. -8
```

will match residues with residue number -8.

xii) Use SMARTS expressions.

From version 7.0 of Maestro it is possible to use SMARTS expressions as part of an ASL expression.

Some examples:

```
smarts. CCC
```

will match all three-carbon subsequences

```
smarts. [R] and atom.ele N
```

will match all ring nitrogens

```
smarts. C1CCCCCC1
```

will match all six-membered carbon rings

4 The Entry Specification Language

This chapter documents the syntax of Maestro's Entry Specification Language (ESL).

4.1 Why an Entry Selection Language?

The Entry Selection Language (ESL) is used for selecting entries in a Maestro project based on the properties of those entries. ESL expressions can be of arbitrary complexity and can use parentheses and logical operators. While the ESL is similar in appearance to the Atom Specification Language (ASL), they are used for quite different purposes - the ASL for specifying sets of atoms in the workspace, the ESL for selecting entries in the project. A typical application for the ESL would be for filtering. An ESL expression can be used to select only those entries that meet a particular criteria, usually based on entry properties. Once selected, those entries can be displayed in the workspace or exported to an external file.

4.2 Entry Properties

There are a number of properties that can be used in ESL expressions:

`entry` The name of the entry is the most useful property at present. The syntax is:

> `entry` ⟨name⟩
> `entry_re` ⟨name_expression⟩

Entry names that contain spaces need to be enclosed in double quotes. The following wildcard characters can be used for the `entry_re` expression:

> `*` matches any number of any characters
> `?` matches any single character
> `#` matches any single digit

Examples:

```
entry anentry
entry crambin-1
entry_re cramb*
entry_re ligand##
```

Note: `name` can be used as an alias for `entry` and `name_re` can be used as an alias for `entry_re`.

`selected` This allows the selection of entries based on whether or not they are already selected in the project. Examples:

```
entry_re lig* and not selected
not selected
```

`included` This allows the selection of entries based on whether or not they are already included for display in the workspace.

```
entry_re lig* and not included
not included
```

`all` This is a quick way to select all entries.

4.3 Logical Operators

The following logical operators are supported (in order of decreasing priority)

- `not`
- `and` `or`

At equal levels of priority expression will be evaluated from left to right. Parentheses can be used to override the default order of priority.

4.4 Entry Property Comparisons

ESL expressions may also contain comparisons involving entry properties. Properties may be defined by their data names or their user names. The data names are the names that appear in the Maestro files. An example is "i_qp_n_stars". User names are the names that appear in the first row of Maestro's Project Table. An example of how this would appear in Maestro's Project Table Header is `n stars`. User names are specified in commands as:

`user(`⟨propertyname⟩`)`

For example:

`user(volume)`

and so,

`user(n stars)` and
`i_qp_n_stars`

are equivalent names for properties.

For integer and real type properties the following comparison operators are supported: `==`, `!=`, `<`, `>`, `<=`, `>=`.

For boolean (logical) properties, the name of the property can just be specified and will match any entries where that property is true. It is also possible to use `==` and `!=` in explicit comparisons with the values `true` and `false`.

Comparisons involving string properties have the form:

⟨property name⟩ ⟨string literal⟩

For example:

```
s_m_string_prop left
s_m_string_prop left*
```

String literals may include the `*`, `?` and `#` wildcard characters available for entry names as described above.

4.5 Examples

Some examples of the use of the ESL are:

```
entry anentry
entry_re ligand*
entry_re lig* and selected
(entry_re lig# and selected ) and not included
user(volume) < 100.0 and selected
included and user(n stars) == 2 and user(n amine) == 0
user(log s) > 0.5 and user(log S) < 1.0
```

5 Commands

1ddataset

Controls the appearance of the a single dataset on a plot of a dihedral drive of a single dihedral.

Syntax:

1ddataset *ccolor*=black | red | green | blue | purple | orange | blue_green | light_green | red_purple | yellow | cyan *curve*=solid | dashed | noline *cwidth*=⟨ n ⟩ *scolor*=black | red | green | blue | purple | orange | blue_green | light_green | red_purple | yellow | cyan *ssize*=⟨ n ⟩ *symbol*=filled_rectangle | hollow_rectangle | filled_circle | hollow_circle | cross | point | filled_diamond | hollow_diamond | no_symbol ⟨ data_set_name ⟩ [⟨ grd_file_name ⟩]

Options:

ccolor The color of the curve used on the graph for this dataset.

Valid values: black, red, green, blue, purple, orange, blue_green, light_green, red_purple, yellow, cyan

Default value: **black**

curve The type of curve which will join the points on the graph.

Valid values: solid, dashed, noline

Default value: **solid**

cwidth The width of the line used on the graph for this dataset.

Valid values: integers

Default value: **1**
Minimum: 1
Maximum: 5

scolor The color of the symbols used on the graph for this dataset.

Valid values: black
red
green
blue
purple
orange
blue_green
light_green
red_purple
yellow
cyan
Default value: **black**

ssize The size of the symbol used on this plot for this dataset.

Valid values: integers
Default value: **3**
Minimum: 2
Maximum: 5

symbol The type of symbol used for this data set on the graph.

Valid values: filled_rectangle
hollow_rectangle
filled_circle
hollow_circle
cross
point
filled_diamond
hollow_diamond
no_symbol
Default value: **filled_rectangle**

Operands:

⟨data_set_name⟩ [⟨grd_file_name⟩]

The first operand is the name of the dataset which is to be created or to have its properties changed. If a second operand is present then that is treated as the name of a file from which the data is to be read. The full name of the file, including the suffix, should be included.

1dplot

Controls the appearance of the plot the energy as a function of a drive of a single dihedral angle.

Syntax:

1dplot *amax*=⟨x⟩ *amin*=⟨x⟩ *emax*=⟨x⟩ *emin*=⟨x⟩
escale=absolute | relative *orientation*=landscape | portrait
papersize=letter | a4 *squareplot*=yes | no *units*=kj | kcal
[⟨postscript_file_name⟩]

Options:

amax The maximum angle value displayed on the graph
Valid values: reals
Default value: **0**

amin The minimum angle value displayed on the graph
Valid values: reals
Default value: **0**

emax The maximum energy value displayed on the graph
Valid values: reals
Default value: **0**

emin The minimum energy value displayed on the graph
Valid values: reals
Default value: **0**

escale The energy scale (absolute/relative) for the graph
Valid values: absolute
relative
Default value: **absolute**

orientation
The paper orientation for the postscript output
Valid values: landscape
portrait
Default value: **landscape**

papersize The paper size for the postscript output
Valid values: letter
a4
Default value: **letter**

squareplot A boolean which controls whether the plot will be constrained to be square.

	Valid values:	boolean (true|false; yes|no; y|n; on|off)
	Default value:	**false**
units	The energy units for the graph	
	Valid values:	kj kcal
	Default value:	**kj**

Operands:

[⟨postscript_file_name⟩]

If an operand is given then this will be treated as the name of a file to which a postscript representation of the plot will be written. The full name of the file, including any suffix, must be included.

1drescale

Rescales the 1D plot axes back to be able to view the entire data range.

Syntax:

1drescale

1dtable

Write out a table of the data from the 1D plot to an external file.

Syntax:

1dtable

2ddataset

Controls the appearance of the dataset plotted on the 2D grid contour plot.

Syntax:

2ddataset *cwidth*=⟨ n ⟩ *emax*=⟨ x ⟩ *emin*=⟨ x ⟩
negativedashed=yes | no *numcontours*=⟨ n ⟩

Options:

cwidth The width of the contours drawn on the plot

Valid values: integers
Default value: **2**
Minimum: 1
Maximum: 5

emax The maximum energy value displayed on the graph

Valid values: reals
Default value: **0**

emin The minimum energy value displayed on the graph

Valid values: reals
Default value: **0**

negativedashed
A boolean which controls whether contours corresponding to negative values will be drawn with a dashed line.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

numcontours
The number of contours used on the graph

Valid values: integers
Default value: **10**
Minimum: 2
Maximum: 10

2dplot

Controls the appearance of a plot of the energy as a function of two dihedral angles.

Syntax:

2dplot *escale*=absolute | relative *orientation*=landscape | portrait *papersize*=letter | a4 *squareplot*=yes | no *units*=kj | kcal [⟨postscript_file_name⟩]

Options:

escale The energy scale (absolute/relative) for the graph

Valid values: absolute
relative
Default value: **absolute**

orientation
The paper orientation for the postscript output

Valid values: landscape
portrait
Default value: **landscape**

papersize The paper size for the postscript output

Valid values: letter
a4
Default value: **letter**

squareplot A boolean which controls whether the plot will be constrained to be square.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

units The energy units for the graph

Valid values: kj
kcal
Default value: **kj**

Operands:

[⟨postscript_file_name⟩]
If an operand is given, this is treated as the name of a file to which a postscript representation of the file is to be written.

2drescale

Rescales the plot axes back to be able to view the entire data range.

Syntax:

2drescale

addfromhold

This command retrieves the structure from the hold set and adds it to the current on-screen structure. The added hold structure is placed at the periphery of the on-screen structure and then a scale-to-screen is done.

Syntax:

addfromhold ⟨hold_name⟩

Operands:
⟨hold_name⟩
The name of the hold. This must be the name which was specified when the hold was created using the "hold" command.

adjustangle

Adjust the bond angle specified by the 3 atoms to the given value.

Syntax:

adjustangle *angle*=⟨x⟩ *move*=attached | terminal | single
⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩

Options:

angle Value to which to set the angle

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	180.0

move This option sets how to move other atoms that attached to the selected moving atom. There are three options: (1) move all attached atoms, (2) move all attached terminal atoms, and (3) move only single atom. Default option is (1).

Valid values:	attached
	terminal
	single

Default value: **attached**

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩

Three atoms which are used to adjust an angle. Note that direction does matter. So a-b-c is different from c-b-a. The third atom 'points' to the part of the structure that will be adjusted.

adjustdihedral

Adjust the dihedral specified by 4 atoms to the given value.

Syntax:

adjustdihedral *dihedral*=⟨ x ⟩ *dihedralterm*=⟨ x ⟩ *move*=attached | terminal | single ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Options:

dihedral Value to which to set the torsion

Valid values: reals
Default value: **0**
Minimum: -180.1
Maximum: 180.1

dihedralterm
Value to which to terminally attached to the dihedral

Valid values: reals
Default value: **0**
Minimum: -180.1
Maximum: 180.1

move This option sets how to move other atoms that attached to the selected moving atom. There are three options: (1) move all attached atoms, (2) move all attached terminal atoms, and (3) move only single atom. Default option is (1).

Valid values: attached
terminal
single
Default value: **attached**

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Four atoms which are used to adjust a torsion. Note that direction does matter. So a-b-c-d is different from d-c-b-a. The fourth atom 'points' to the part of the structure that will be adjusted.

adjustdistance

Adjust the distance specified by 2 atoms to the given value.

Syntax:

adjustdistance *distance*=⟨ x ⟩ *move*=attached | terminal | single
⟨ atom1 ⟩ ⟨ atom2 ⟩

Options:

distance Value to which to set the torsion

Valid values:	reals
Default value:	**0**
Minimum:	0.0

move This option sets how to move other atoms that attached to the selected moving atom. There are three options: (1) move all attached atoms, (2) move all attached terminal atoms, and (3) move only single atom. Default option is (1).

Valid values:	attached terminal single
Default value:	**attached**

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩

Two atoms which are used to adjust a distance. Note that direction does matter. So a-b is different from b-a. The second atom 'points' to the part of the structure that will be adjusted.

alias

Define an alias for a command. This allows an abreviated symbol to be defined for any command.

Syntax:

alias ⟨alias_name⟩ ⟨definition⟩

Operands:

⟨alias_name⟩ ⟨definition⟩

The first operand is the name of the alias. If this contains embedded spaces then it must be enclosed in double quotes. The remaining operands are the definition of the alias. When command processing takes place, all occurences of the alias name (the first operand) will be replaced by the definition of the alias.

angle

Specifies a triplet of atoms to have their bond angle measured and displayed.

Syntax:

angle ⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩

Operands:

⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩

The three atoms between which the angle is to be measured. Note that the specifying a-b-c is the same as specifying c-b-a

appendribbons

Generate a ribbon for the parts that are not currently in ribbon.

Syntax:

appendribbons

application

Determines which backend application is currently active and which application-specific menu is displayed. For example the command application Impact will result in the Impact menu being displayed in the main menu bar.

Syntax:

application ⟨application_name⟩

Operands:

⟨application_name⟩

The name of the backend application which Maestro is currently interacting with. Current options are "none", "macromodel" or "impact".

atom

Create a new atom in space with the current type and at the position specified by the operands.

Syntax:

atom *by*=element | type *element*=⟨text⟩ *type*=⟨n⟩ ⟨x⟩ ⟨y⟩ ⟨z⟩

Options:

by This options determines whether atoms are to be described by element symbols (the element= option) or by atom types (type= option).

Valid values: element
type
Default value: **element**

element This option sets the current element of all atoms to be created (or retyped) if the by option == element

Valid values: text strings
Default value: **C**

type This option sets the current type of all atoms to be created (or retyped) if the by option == type

Valid values: integers
Default value: **3**

Operands:
⟨x⟩ ⟨y⟩ ⟨z⟩
The operands are three real numbers which specify the x, y and z coordinates for the new atom.

atomname

Set the PDB atom name for all atoms which match the ASL specification.

Syntax:

atomname ⟨PDBNAME⟩ ⟨ASL⟩

Operands:
⟨PDBNAME⟩ ⟨ASL⟩
The first operand is the PDB atom name which will be used for all atoms which match the specification. The second operand is a valid ASL string which defines the set of atoms which are to have their atom names changed.

attach

Attach the currently selected fragment to the specified atom. The bond to this atom (and the atom itself) will be replaced by a bond from the incoming fragment.

Syntax:

attach ⟨atom_num⟩

Operands:
⟨atom_num⟩
The atom number defines the bond to which the fragment be attached. The atom specifed must be a terminally attached atom.

autosetup

Performs the automatic setup for the conformational search. The setup is only performed on those atoms which match the ASL definition given in the operand.

Syntax:

autosetup *chiralatoms*=yes | no *compatoms*=yes | no
rings=yes | no *torsionchecks*=yes | no ⟨ASL⟩

Options:

chiralatoms
A boolean option which determines whether the the automatic setup will include finding chiral atoms.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

compatoms
A boolean option which determines whether the the automatic setup will include finding comparison atoms.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

rings
A boolean option which determines whether the the automatic setup will include finding ring closures.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

torsionchecks
A boolean option which determines whether the the automatic setup will include finding torsion angles to be checked.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
⟨ASL⟩
A string in the atom specification language. The automatic setup will be peformed for only that atoms which match this ASL specification.

beginundoblock

Begin a new undoable command block.

Syntax:

beginundoblock

bmincomfile

Write a MacroModel command file with the current energy settings.

Syntax:

bmincomfile

bond

Create a new bond between two atoms or increments the bond order between two existing atoms.

Syntax:

bond ⟨atom1⟩ ⟨atom2⟩

Operands:
⟨atom1⟩ ⟨atom2⟩
The operands are the numbers of the two atoms which are to be connected by a bond or have the bond order of an existing bond incremented.

bondorder

Increment, decrement or set the bond order for the specified bond.

Syntax:

bondorder increment|decrement|⟨bond_order⟩ ⟨atom1⟩ ⟨atom2⟩

Operands:
increment|decrement|⟨bond_order⟩ ⟨atom1⟩ ⟨atom2⟩

The first operand is either the word "increment", the word "decrement", or an integer value representing the bond order to be applied to the specifed bond. Note that "0" is a valid bond order. The second and third operands are the atoms which define the bond which is to have its bond order changed.

bondtonew

Create a new atom and bond it with a single bond to an existing atom.

Syntax:

bondtonew ⟨ atom_num ⟩ ⟨ x ⟩ ⟨ y ⟩ ⟨ z ⟩

Operands:
⟨ atom_num ⟩ ⟨ x ⟩ ⟨ y ⟩ ⟨ z ⟩
The first operand is the number of an existing atom from which the bond is to be drawn. The remaining three operands are the x, y and z coordinates at which the new atom is to be placed.

buildoptions

Sets options associated with structure building. These are persistent and will be retained between Maestro sessions.

Syntax:

buildoptions *adjustbondlengths*=yes | no
adjustnumhydrogens=yes | no *useunitedatoms*=yes | no

Options:

adjustbondlengths
When an atom type is changed, bond lengths around that atom will be set to ideal values.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

adjustnumhydrogens
During bond order, drawing or formal charge change operations, rectify the number of hydrogens to be consistent with the new bonding or formal charge.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

useunitedatoms
Will allow the generation of united atom types while structure building.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

calcenergy

Used to set options associated with a single point energy calculation.

Syntax:

calcenergy *listing*=none | minimal | complete

Options:

listing This option determines the extent of the listing of energy components.

Valid values: none
minimal
complete
Default value: **none**

Aliases:
ecalc (see [ecalc], page 82)

canonicalname

Sets atom names to one generated from atom properties (such as element or PDB names) for all the atoms matched by the ASL operand.

Syntax:

canonicalname ⟨separator⟩ ⟨option_flag⟩ ⟨ASL⟩

Operands:
⟨separator⟩ ⟨option_flag⟩ ⟨ASL⟩

The first operand is the separator symbol to be used in generating the canonical names of all atoms. The second operand is the option_flag, which indicates which two atom properties will be used to construct the canonical names. Possible values are "3" - from element and atom number and "12" PDB chain, residue and atom name. The final operand is the ASL expression to indicate which atoms are to have a canonical name set.

cascadepanels

Cascade visible panels

Syntax:

cascadepanels

cd

This is a standard alias for **changedirectory** (see [changedirectory], page 51).

cellsmarkerdump

Print out the current option values of the Glide hydrophobic Cells markers command.

Syntax:

cellsmarkerdump

cellsmarkersettings

Set graphical data of Glide hydrophobic cells markers.

Syntax:

cellsmarkersettings *activeblue*=⟨x⟩ *activegreen*=⟨x⟩ *activered*=⟨x⟩ *ambient*=⟨x⟩ *blue*=⟨x⟩ *diffuse*=⟨x⟩ *emission*=⟨x⟩ *green*=⟨x⟩ *highlightblue*=⟨x⟩ *highlightgreen*=⟨x⟩ *highlightred*=⟨x⟩ *labelregions*=yes | no *red*=⟨x⟩ *regionblue*=⟨x⟩ *regiongreen*=⟨x⟩ *regionred*=⟨x⟩ *roundingeffect*=⟨x⟩ *shininess*=⟨x⟩ *specular*=⟨x⟩ *step*=⟨n⟩ *transparency*=⟨x⟩

Options:

activeblue The blue color component of cell active frame and region label.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

activegreen

The green color component of cell active frame and region label.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

activered The red color component of cell active frame and region label.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

ambient Set material property - ambient, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

blue The blue color component of cell markers.

Valid values: reals
Default value: **0.75**
Minimum: 0.0
Maximum: 1.0

diffuse Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.5**
Minimum: 0.0

Maximum: 1.0

emission Set material property - emission, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

green The green color component of cell markers.

Valid values: reals
Default value: **0.75**
Minimum: 0.0
Maximum: 1.0

highlightblue
The blue color component of cell markers if the cell is of a highlight region.

Valid values: reals
Default value: **0.3**
Minimum: 0.0
Maximum: 1.0

highlightgreen
The green color component of cell markers if the cell is of a highlight region.

Valid values: reals
Default value: **0.3**
Minimum: 0.0
Maximum: 1.0

highlightred
The red color component of cell markers if the cell is of a highlight region.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

labelregions
This option determines whether a region should be labeled (true) or not (false).

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

red The red color component of cell markers.

Valid values: reals

Default value: **0.75**
Minimum: 0.0
Maximum: 1.0

regionblue The blue color component of cell markers if the cell is of a region.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

regiongreen

The green color component of cell markers if the cell is of a region.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

regionred The red color component of cell markers if the cell is of a region.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

roundingeffect

This determines the rounding effect of edges of a cell, for front face.

Valid values: reals
Default value: **8**
Minimum: 0.0

shininess Set material property - shininess, for front face.

Valid values: reals
Default value: **80**
Minimum: 0.0
Maximum: 128.0

specular Set material property - specular, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

step The step domain tolerance of cells.

Valid values: integers
Default value: **3**
Minimum: 1

transparency
The transparency of phobic cell markers.

Valid values:	reals
Default value:	**30**
Minimum:	0.0
Maximum:	100.0

centeratom

Set global center of rotation to the given atom

Syntax:

centeratom *atom*=⟨ n ⟩

Options:

atom Atom to center transformations on

Valid values:	integers
Default value:	**1**
Minimum:	1

centerbond

Set global center of rotation to the given bond

Syntax:

centerbond *at1*=⟨ n ⟩ *at2*=⟨ n ⟩

Options:

at1 Atom1 of bond to be center of transformation

Valid values:	integers
Default value:	**1**
Minimum:	1

at2 Atom 2 of bond to be center of transformation

Valid values:	integers
Default value:	**1**
Minimum:	1

centercoordinates

Set global center of rotation to the given coordinate

Syntax:

centercoordinates *x*=⟨x⟩ *y*=⟨x⟩ *z*=⟨x⟩

Options:

x X center

Valid values: reals
Default value: **0**

y Y center

Valid values: reals
Default value: **0**

z Z center

Valid values: reals
Default value: **0**

centroid

This command takes the previously defined centroid atoms, averages the x, y, z co-ordinates and then creates a new atom of type 61 at the average position.

Syntax:

centroid

centroidatom

Define a single atom which will be used to define the centroid. Associated with this atom is a single atom marker.

Syntax:

centroidatom ⟨atom_num⟩

Operands:

⟨atom_num⟩

The number of the atom to be used to define the centroid.

chainname

Set the chain name for all atoms which match the ASL specification.

Syntax:

chainname ⟨CHAINNAME⟩ ⟨ASL⟩

Operands:

⟨CHAINNAME⟩ ⟨ASL⟩

The first operand is the PDB chain name (a single character) that will be used for all atoms which match the specification. The second operand is a valid ASL string which defines the set of atoms which are to have their residue names changed.

changedirectory

Change the current directory to that given by the operand of this command.

Syntax:

changedirectory ⟨new_directory⟩

Operands:

⟨new_directory⟩

The name of the directory to be made the current directory.

Aliases:

cd (see [cd], page 45), **chdir** (see [chdir], page 52)

chdir

This is a standard alias for **changedirectory** (see [changedirectory], page 51).

chiralatom

Specifies an atom which is to be marked as "chiral" in the conformational search

Syntax:

chiralatom ⟨atom_number⟩

Operands:

⟨atom_number⟩

The number of an atom which is to be added to the list of atoms which are to be considered as chiral in the conformational search.

clip

Adjust the front and/or back clipping planes

Syntax:

clip *back*=⟨x⟩ *front*=⟨x⟩

Options:

back Position at which to set the back clipping plane

Valid values: reals
Default value: **0**

front Position at which to set the front clipping plane

Valid values: reals
Default value: **0**

clusteratom

Defines a single comparison atom to be used for XCluster jobs where clustering is based on Atomic RMS differences.

Syntax:

clusteratom ⟨atom⟩

Operands:

⟨atom⟩

The number of an atom which is to added to the list of comparison atoms for a clustering calculation.

clusterheavyatoms

Specifies the atoms to be used in comparisons of conformers in XCluster jobs where clustering is based on Atomic RMS differences. The atoms specified are all of the heavy atoms and optionally hydrogens on oxygen and sulfur.

Syntax:

clusterheavyatoms *osh*=yes | no

Options:

osh A boolean which controls whether hydrogens on oxygen and sulfur are included with heavy atoms.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

clusterset

Specifies a set of atoms to be used in comparisons of conformers in XCluster jobs where clustering is based on Atomic RMS differences.

Syntax:

clusterset ⟨ASL⟩

Operands:
⟨ASL⟩
A string in the atom specification language. All atoms which match this description will be added to the list of XCluster comparison atoms.

clustertorsion

Specifies four atoms which define a torsion to be compared in XCluster jobs where clustering is based on Torsional RMS differences.

Syntax:

clustertorsion ⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩

Operands:
⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩
The numbers of four atoms which define a torsion angle to be compared in XCluster jobs. Note that specifying a-b-c-d is the same as specifying d-c-b-a.

coloratom

Set the color for a group of atoms defined by the ASL operand.

Syntax:

coloratom *cindex*=⟨n⟩ *color*=⟨text⟩ ⟨ASL⟩

Options:

cindex An integer which indicates color index which is to be used for the atoms

Valid values:	integers
Default value:	**2**
Minimum:	1
Maximum:	128

color A string which is the color name for atom coloring. Valid color names are described in the file $SCHRODINGER/maestro-vX.X/data/res/colors.res

Valid values: text strings
Default value:

Operands:

⟨ ASL ⟩

A string in the atom specification language. All atoms which match this specification will have their color changed to the current color.

colorscheme

Apply a predefined color scheme to the group of atoms defined by the ASL spec.

Syntax:

colorscheme *scheme*=⟨ text ⟩ ⟨ ASL ⟩

Options:

scheme The name of the current color scheme.

Valid values: text strings
Default value: **atype**

Operands:

⟨ ASL ⟩

A string in the atom specification language. All atoms which match this specification will have their color changed to match the currently selected color scheme.

compareatom

Defines a single comparison atom to be used during a multiple minimization or conformational search. Comparison atoms are used to make comparisons in the process of determining if conformers are unique.

Syntax:

compareatom ⟨atom⟩

Operands:
⟨atom⟩
The number of an atom which is to added to the list of comparison atoms during a conformational search.

compareset

Specifies a set of atoms to be used in comparisons of conformers during a multiple minimization or conformational search.

Syntax:

compareset ⟨ASL⟩

Operands:
⟨ASL⟩
A string in the atom specification language. All atoms which match this description will be added to the list of comparison atoms.

confelim

This keyword is used to set various options associated with starting Redundant Conformer Elimination jobs from Maestro.

Syntax:

confelim *compare_in_place*=yes | no *energy_source*=none | jaguar | mm2* | mm3* | amber* | opls* | amber94 | mmff | mmffs | oplsaa | opls2003 *incorporate*=append | replace | ignore

Options:

compare_in_place
A boolean which controls whether ConfElim will compare structures without first doing a superposition.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

energy_source

Source of energy for the comparisons

Valid values: none
jaguar
mm2*
mm3*
amber*
opls*
amber94
mmff
mmffs
oplsaa
opls2003

Default value: **mmffs**

incorporate

How the results are to be incorporated into the project. This can be done with replacement of the existing entries or by appending as new entries to the project or by ignoring the final results.

Valid values: append
replace
ignore

Default value: **append**

confelimstart

Start a Redundant Conformer Elimination job with the current settings.

Syntax:

confelimstart

confelimwrite

Write a Redundant Conformer Elimination input file with the current settings.

Syntax:

confelimwrite

confgenltsearch

Defines settings for ConfGen conformational searching in MacroModel.

Syntax:

confgenltsearch *amidebonds*=vary | retain | trans
compareatoms=none | heavy | heavy_polar_h
distinguishenantiomers=yes | no *limitsave*=yes | no
maxdist=⟨ x ⟩ *maxringconf*=⟨ n ⟩ *maxtorsdiff*=⟨ x ⟩
numsave=⟨ n ⟩ *numsteps*=⟨ n ⟩ *samplerings*=yes | no
searchmode=standard | rapid | complete | thorough
window=⟨ x ⟩

Options:

amidebonds
This determines whether to vary amide bond conformation, retain original amide bond conformation, or set amide bond conformation to trans.
Valid values: vary
retain
trans
Default value: **vary**

compareatoms
This determines which kinds of comparison atoms are automatically identified for judging redundant conformers.
Valid values: none
heavy
heavy_polar_h
Default value: **heavy_polar_h**

distinguishenantiomers
A boolean for whether to retain different enantiomers. If true, an additional opcode, NANT, is written to the .com file.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

limitsave A boolean which controls whether number of structures saved is limited by numsave option (if true) or by numsteps (if false).
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

maxdist Maximum distance between atoms in equal structures.

Valid values: reals
Default value: **0.5**
Minimum: 0.0

maxringconf
The maximum number of ring conformations generated by the search.

Valid values: integers
Default value: **8**
Minimum: 1

maxtorsdiff
Maximum torsional angle difference between polar hydrogens in equal structures.

Valid values: reals
Default value: **60**
Minimum: 0.0
Maximum: 180.0

numsave The number of structures that will be saved at the end of each search.

Valid values: integers
Default value: **1000**
Minimum: 0

numsteps An option which sets the number of steps which will be performed during the ConfGen conformational search.

Valid values: integers
Default value: **1000**
Minimum: 0

samplerings
A boolean which controls whether to sample rings.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

searchmode
This determines whether rapid (standard) or thorough (complete) search will be used.

Valid values: standard
rapid
complete
thorough
Default value: **complete**

window The energy window (in kJ/mol) within which structures will be saved.

Valid values: reals
Default value: **25**
Minimum: 0.0

confgenmini

Used to set values associated with a MacroModel energy minimization for Ligand Torsional Search (ConfGen)

Syntax:

confgenmini *converge*=nothing | energy | gradient | movement
method=sd | prcg | osvm | fmnr | tncg | lbfgs
postmaxiter=⟨ n ⟩ *premaxiter*=⟨ n ⟩ *threshold*=⟨ x ⟩

Options:

converge This option determines which convergence criterion will be used during an energy minimization.

Valid values: nothing
energy
gradient
movement
Default value: **gradient**

method This option determines which minimization method will be used.

Valid values: sd
prcg
osvm
fmnr
tncg
lbfgs
Default value: **tncg**

postmaxiter
This option determines the maximum number of iterations for post-minimization of generated structures.

Valid values: integers
Default value: **50**
Minimum: 0
Maximum: 9999999

premaxiter
This option determines the maximum number of iterations for pre-minimization of input structures.

Valid values: integers
Default value: **100**
Minimum: 0
Maximum: 9999999

threshold This option determines what the convergence threshold will be.

Valid values: reals
Default value: **0.05**
Minimum: 0.0

confgenpotential

Set various options associated with the definition of the potential energy to be used in a MacroModel job.

Syntax:

confgenpotential *cele*=⟨x⟩ *charges*=force_field | structure_file *chnd*=⟨x⟩ *cutoff*=normal | extended | user_defined | none *cvdw*=⟨x⟩ *dielectric*=⟨x⟩ *electrostatics*=field_field | constant | distance_dependant *field*=mm2* | mm3* | amber* | opls* | amber94 | mmff | mmffs | oplsaa | opls2003 *solvent*=none | water | chcl3 | octanol *suppresshbond*=yes | no

Options:

cele This option determines what cutoff will be used for the electrostatic part of the energy calculation.

Valid values: reals
Default value: **12**
Minimum: 0.0
Maximum: 99999.0

charges This option determines where the charges to be used in the energy calculation will come from.

Valid values: force_field
structure_file
Default value: **force_field**

chnd This option determines what cutoff will be used for the hydrogen bond part of the energy calculation.

Valid values: reals
Default value: **4**

Minimum: 0.0
Maximum: 99999.0

cutoff This option determines what type of non-bonded cutoff will be used in the energy calculation.

Valid values: normal
extended
user_defined
none
Default value: **normal**

cvdw This option determines what VDW cutoff will be used in the energy calculation.

Valid values: reals
Default value: **7**
Minimum: 0.0
Maximum: 99999.0

dielectric The dielectric constant to be used in the electrostatic part of the energy calculation.

Valid values: reals
Default value: **4**
Minimum: 0.9999999999

electrostatics
The electrostatic treatment to be used in the energy calculation.

Valid values: field_field
constant
distance_dependant
Default value: **distance_dependant**

field The force field to be used for the energy calculation.

Valid values: mm2*
mm3*
amber*
opls*
amber94
mmff
mmffs
oplsaa
opls2003
Default value: **mmffs**

solvent The solvent model to be used for the energy calculation

Valid values: none
water
chcl3
octanol

Default value: **none**

suppresshbond
A boolean which controls whether to suppress hydrogen bonding electrostatics.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

confgenreadpotential

Read potential settings from a command file.

Syntax:

confgenreadpotential ⟨file_name⟩

Operands:
⟨file_name⟩
The name of the file from which the potential settings will be read. If no name is specified, the default settings will be used.

confgenstart

Start a Conformer Generation (Ligand Torsion Search) job with the current settings.

Syntax:

confgenstart

confgenwrite

Write a Conformer Generation (Ligand Torsion Search) input file with the current settings.

Syntax:

confgenwrite

confsearch

Defines settings for conformational searching in MacroModel.

Syntax:

confsearch *distinguishenatiomers*=yes | no *enableauto*=yes | no *max_distance_lmcs*=⟨ x ⟩ *maxdist*=⟨ x ⟩ *method*=mcmm | serial_mcmm | summ | mixed_lmcs | serial_mcmm_lmcs | pure_lmcs | serial_lmcs | large_lmcs | mixed_large_lmcs *min_distance_lmcs*=⟨ x ⟩ *numsteps*=⟨ n ⟩ *numstructures*=⟨ n ⟩ *probability_tors_lmcs*=⟨ x ⟩ *window*=⟨ x ⟩

Options:

distinguishenatiomers
This option determines to conside conformers more than once

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

enableauto Toggle whether or not to use AUTO setup for processing multiple structures.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

max_distance_lmcs
The maximum possible distance for the fastest moving atom in each LMCS move.

Valid values: reals
Default value: **6**
Minimum: 0.0

maxdist Maximum distance between atoms in equal structures.

Valid values: reals
Default value: **0.25**
Minimum: 0.0

method This determines which method will be used to perform the conformational search.

Valid values: mcmm
serial_mcmm
summ
mixed_lmcs
serial_mcmm_lmcs
pure_lmcs
serial_lmcs
large_lmcs
mixed_large_lmcs
Default value: **mcmm**

min_distance_lmcs
The minimum possible distance for the fastest moving atom in each LMCS move.

Valid values: reals
Default value: **3**
Minimum: 0.0

numsteps An option which sets the number of steps which will be performed during the conformational search.

Valid values: integers
Default value: **1000**
Minimum: 0

numstructures
The number of structures that will be saved at the end of each search.

Valid values: integers
Default value: **0**
Minimum: 0

probability_tors_lmcs
The probability that a TORS/MOLS move will be made during an LMCS search.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

window The energy window (in kJ/mol) within which structures will be saved.

Valid values: reals
Default value: **50**
Minimum: 0.0

connect

Connect atom pairs.

Syntax:

connect

connectfuseatom

Define an atom pair for which will be connected in a subsequent connect operation.

Syntax:

connectfuseatom ⟨atom1⟩ ⟨atom2⟩

Operands:
⟨atom1⟩ ⟨atom2⟩
Two atom numbers which represent an atom pair to be connected by a subsequent connect or fuse command. The two atoms must be from different molecules and all connectfuseatom commands must specify molecules in the same order.

constrainedangle

Specifies a angle between three atoms which is to be constrained by a harmonic constraint during a MacroModel calculation.

Syntax:

constrainedangle *angle*=⟨x⟩ *constant*=⟨x⟩ *width*=⟨x⟩ ⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩

Options:

angle The angle at which the atoms are to be constrained.
Valid values: reals
Default value: **-1**

constant The force constant for the harmonic constraint to be applied to the angle.

Valid values: reals
Default value: **100**
Minimum: 0.0

width The half-width for a flat-bottomed constraint.

Valid values: reals
Default value: **0**
Minimum: 0.0

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩
Three atoms which are to have the angle between them constrained by a harmonic constraint. The three atoms are not necessarily connected. Note that specifying a-b-c will be treated by the program as the same as specifying c-b-a.

constrainedatom

Specifies a single atom to be constrained at its current positions during a MacroModel energy calculation. The atom will be constrained by the use of a harmonic constraint.

Syntax:

constrainedatom *constant*=⟨ x ⟩ *width*=⟨ x ⟩ ⟨ atom_number ⟩

Options:

constant The force constant (in kJ/mol/Angs) for the harmonic constraint.

Valid values: reals
Default value: **0**

width The half-width of a flat bottomed restraint.

Valid values: reals
Default value: **0**
Minimum: 0

Operands:
⟨ atom_number ⟩

The number of the atom to which a constraint is to be applied.

constraineddist

Specifies a distance between two atoms which is to be constrained by a harmonic constraint during a MacroModel calculation.

Syntax:

constraineddist *constant*=⟨x⟩ *distance*=⟨x⟩ *remove_nb*=yes | no *width*=⟨x⟩ ⟨atom1⟩ ⟨atom2⟩

Options:

constant The force constant for the harmonic constraint to be applied to the distance.

Valid values: reals
Default value: **100**
Minimum: 0.0

distance The distance at which the atoms are to be constrained.

Valid values: reals
Default value: **-1**

remove_nb A boolean option which controls whether the non-bonded interaction (if any) between the two specified atoms will be removed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

width The half-width for a flat-bottomed constraint.

Valid values: reals
Default value: **0**
Minimum: 0.0

Operands:

⟨atom1⟩ ⟨atom2⟩

Two atoms which are to have the distance between them constrained by a harmonic constraint Note that specifying a-b will be treated by the program as the same as specifying b-a.

constrainedset

Specifies a set of atoms to be constrained at their current positions. The atoms will be constrained by the use of a harmonic constraint.

Syntax:

constrainedset *constant*=⟨ x ⟩ *width*=⟨ x ⟩ ⟨ ASL ⟩

Options:

constant The force constant (in kJ/mol/Angs) for the harmonic constraint.

Valid values:	reals
Default value:	**100**
Minimum:	0.0

width The half-width of a flat bottomed restraint

Valid values:	reals
Default value:	**0**
Minimum:	0

Operands:
⟨ ASL ⟩
The operand must be a valid string in the atom specification language. Any atoms which match will have constraints generated for them.

constrainedtorsion

Specifies a torsion between four atoms which is to be constrained by a harmonic constraint during a MacroModel calculation.

Syntax:

constrainedtorsion *constant*=⟨ x ⟩ *torsion*=⟨ x ⟩ *width*=⟨ x ⟩
⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Options:

constant The force constant for the harmonic constraint to be applied to the torsion.

Valid values:	reals

Default value: **100**
Minimum: 0.0

torsion The torsion at which the atoms are to be constrained.

Valid values: reals
Default value: **500**

width The half-width for a flat-bottomed constraint.

Valid values: reals
Default value: **0**
Minimum: 0.0

Operands:

⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩

Four atoms which are to have the angle between then constrained by a harmonic constraint. Note that specifying a-b-c-d is treated by the program the same as specifying d-c-b-a.

contactcriteria

Specify the criteria for calculating good, bad, ugly contacts. There is no upper-bound to C. C = {dist between the two atoms} divided by {vdW radius first atom + vdW radius of second atom}.

Syntax:

contactcriteria *bad*=⟨x⟩ *displaybad*=yes | no *displaygood*=yes | no *displayugly*=yes | no *good*=⟨x⟩ *ugly*=⟨x⟩

Options:

bad Criteria for bad contact

Valid values: reals
Default value: **0.89**
Minimum: 0.0

displaybad If contact markers are shown, this option determines whether the bad contact markers will be displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

displaygood
If contact markers are shown, this option determines whether the good contact markers will be displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

displayugly
If contact markers are shown, this option determines whether the ugly contact markers will be displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

good Criteria for good contact

Valid values: reals
Default value: **1.3**
Minimum: 0.0

ugly Criteria for ugly contact

Valid values: reals
Default value: **0.75**
Minimum: 0.0

contactset1

Specify the first set of atoms used in determining good, bad, and ugly contacts.

Syntax:

contactset1 ⟨ASL⟩

Operands:

⟨ASL⟩

A string in the atom specification language. Typical usage is to define contactset1 and contactset2. This set, contactset1, defines the "from" atoms. The contactset2 atoms define the "to" atoms. Contacts are calculated between these two sets. That is, the contacts are inter-set contacts. No intra-set contacts are calculated. If contactset2's ASL string is empty, then contacts are calculated for all atoms in contactset1.

contactset2

Specify the second set of atoms used in determining good, bad, and ugly contacts.

Syntax:

contactset2 ⟨ASL⟩

Operands:

⟨ASL⟩

A string in the atom specification language. Typical usage is to define contactset1 and contactset2. This set, contactset2, defines the "to" atoms. The contactset1 atoms define the "from" atoms. Contacts are calculated between these two sets. That is, the contacts are inter-set contacts. No intra-set contacts are calculated. If contactset2's ASL string is empty, then contacts are calculated for all atoms in contactset1.

createpropsubset

This command creates a property subset for the given table using the given property names.

Syntax:

createpropsubset ⟨table⟩ ⟨propertynames⟩

Operands:

⟨table⟩ ⟨propertynames⟩

The number of the table to create the subset for. If the table operand is missing, no resize will be done. The names of the properties to use to create the subset.

createsubset

Creates a subset in the project table using the currently selected entries. This function also switches to subset view.

Syntax:

createsubset

defaultfc

Requires a single operand which is the default force constant to be used for the constrainedatom command. The default value is used only when no value has been specified by the constant= option of the "constrainedatom" command.

Syntax:

defaultfc ⟨default_force_constant_value⟩

Operands:
⟨default_force_constant_value⟩
The value of the default force constant in kJ/mol/Angstrom

delete

Delete a named object. The object type is the same as the command which is used to create that type of object. For example to delete a set named "set1" use: delete set set1. All instances of any type of object can be deleted with the "all" name. There are two special objects: "atom" and "bond" for deleting atoms and bonds respectively
For example: delete set set1
delete constrainedatom 1
delete set all delete res. 1-5 delete bond 1 2

Syntax:

delete *includeterminal*=yes | no ⟨object_type⟩
⟨object_name⟩|all|⟨ASL⟩|⟨bond⟩

Options:

includeterminal
This option is used to control the behavior when deleting atoms and bonds. If this is "true" then all terminally attached atoms will also be deleted when deleting bonds and atoms.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

Operands:

⟨object_type⟩ ⟨object_name⟩|all|⟨ASL⟩|⟨bond⟩

The first operand determines what type of object is to be deleted. This can be the name of any of the named objects in Maestro (sets, filters, holds etc.) or can be "atom" or "bond". The subject operands depend on the first operand. If the first operand names an object then the second operand must be the name which was specified when that object was created. If the first operand is "atom" then the second operand must be a valid string in the atom specification language and all atoms which match that string will be deleted. Finally if the first operand is "bond" then the second and third operands are expected to be the numbers of the two atoms which define that bond.

deleteproperty

This is a standard alias for **propertydelete** (see [propertydelete], page 321).

dihedral

Specifies a quartet of atoms to have their dihedral angle measured and displayed.

Syntax:

dihedral ⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩

Operands:

⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩

The four atoms between which the dihedral angle is to be measured. Note that the specifying a-b-c-d is the same as specifying d-c-b-a

dihedraldrive

Used to set a dihedral angle to be driven. The four operands are the atom numbers defining the dihedral to be driven. A maximum of two angles can be driven.

Syntax:

dihedraldrive *finish*=⟨ x ⟩ *increment*=⟨ x ⟩ *start*=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Options:

finish Specifies the finishing angle for the dihedral drive.
Valid values: reals
Default value: **360**

increment Specifies the angle increment for the dihedral drive.
Valid values: reals
Default value: **30**

start Specifies the starting angle for the dihedral drive.
Valid values: reals
Default value: **0**

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩
The four operands are treated as atom numbers which define the dihedral to be driven.

displayatom

Display all atoms in the set described by the ASL operand.

Syntax:

displayatom ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language which describes the set of atoms which are to be displayed.

displayonlyatom

Display only the set described by the ASL operand.

Syntax:

displayonlyatom ⟨ASL⟩

Operands:

⟨ASL⟩

A string in the atom specification language which describes the set of atoms which are to be displayed.

displayonlypose

This command undisplays all poses in the pose viewer, then displays the specified structure from the pose set.

Syntax:

displayonlypose ⟨pose_name⟩

Operands:

⟨pose_name⟩

The name of the pose structure.

displayopt

Set properties for the display, such as the background color.

Syntax:

displayopt *adjustclip*=yes | no *aligndepth*=⟨ x ⟩ *bgcindex*=⟨ n ⟩ *bgcolor*=⟨ text ⟩ *eye_sepf*=⟨ x ⟩ *fog*=auto | on | true | yes | off | false | no *fogatomlabels*=yes | no *fogcutoff*=⟨ n ⟩ *fogdensity*=⟨ x ⟩ *fogendz*=⟨ x ⟩ *fogstartz*=⟨ x ⟩ *fogtype*=linear | exp | exp2 *maximize*=yes | no *perspective*=yes | no *perspectivescale*=⟨ x ⟩ *sepf*=⟨ x ⟩ *sizef*=⟨ x ⟩ *start_str*=⟨ text ⟩ *stereo*=yes | no *stereomethod*=hardware | crosseyed | walleyed | fullscreen *stop_str*=⟨ text ⟩

Options:

adjustclip A boolean option which determines whether to adjust clipping to preserve stereo depth.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

aligndepth Relative z-depth (1 for back clipping plane, 0 for front clipping plane) at which stereo images appear to converge. Moves region inside clipping planes in or out of the screen.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

bgcindex An integer which indicates color index which is to be used for the background

Valid values: integers
Default value: **1**
Minimum: 1
Maximum: 128

bgcolor A string which is the color name for backgound color. Valid color names are described in the file $SCHRODINGER/maestro-vX.X/data/res/colors.res

Valid values: text strings
Default value:

eye_sepf Eye separation for stereo as fraction of normal. Actual separation may be made smaller to limit maximum stereo displarity to about 1 inch at clipping planes.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 2.0

fog Set fog to be displayed, to not be displayed, or to toggle automatically based on the number of atoms in the workspace.

Valid values: auto
on
true
yes
off
false
no
Default value: **auto**

fogatomlabels
A Boolean option which determines whether to display fog on atom labels (only displayed if fog is already on in general).

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

fogcutoff Specifies the minimum number of atoms in the workspace for which fog will be displayed, when fogging is set to auto.

Valid values: integers
Default value: **40**
Minimum: 0

fogdensity Factor controlling optical density of fog.

Valid values: reals
Default value: **1.5**
Minimum: 0.0
Maximum: 4.0

fogendz Factor controlling end of linear fog ramp.

Valid values: reals
Default value: **1**
Minimum: 1.0
Maximum: 2.0

fogstartz Factor controlling start of linear fog ramp.

Valid values: reals
Default value: **0.35**
Minimum: 0.0
Maximum: 0.999

fogtype Specifies method used for calculating fog.

Valid values: linear
exp
exp2
Default value: **linear**

maximize A boolean option which determines whether to maximize the display window (on) or restore to the original window size (off)

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

perspective
A boolean option which determines whether to do perspective, rather than othogonal, projection.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

perspectivescale
Ratio between the actual size and the projected size of an object placed 1 foot behind display screen. This has a corresponding viewing distance, view_dist = (1 ft) / (perspectivescale - 1). For a known viewing distance, perspectivescale = 1 + (1 ft / view_dist). Therefore, for a normal viewing distance of 2 feet, perspectivescale = 1 + (1 ft / 2 ft) = 1.5.

Valid values: reals
Default value: **1.5**
Minimum: 1.0
Maximum: 4.0

sepf Image separation as fraction of maximum horizontal separation (window width minus image width)

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

sizef Image size as fraction of available width

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

start_str The system command to start hardware stereo.

Valid values: text strings
Default value:

stereo A boolean option which determines whether to display in stereo

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

stereomethod
Specifies method used for stereo viewing

Valid values: hardware
crosseyed
walleyed
fullscreen

Default value: **hardware**

stop_str The system command to stop hardware stereo.

Valid values: text strings
Default value:

displaypose

This command retrieves the pose set structure in the pose viewer and adds it to the current on-screen structure, after removing atoms belonging to the pose structure from the workspace.

Syntax:

displaypose ⟨pose_name⟩

Operands:
⟨pose_name⟩
The name of the pose structure.

distance

Specifies a pair of atoms to have their distance measured and displayed.

Syntax:

distance ⟨atom1⟩ ⟨atom2⟩

Operands:
⟨atom1⟩ ⟨atom2⟩
The two atoms between which the distance is to be measured. Note that the specifying a-b is the same as specifying b-a

distancecheck

Specifies two atoms which define a distance to be checked during a conformational search.

Syntax:

distancecheck *allowed*=⟨ x ⟩ *value*=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩

Options:

allowed The maximum allowed variation from the "target" value during the distance check.

Valid values: reals
Default value: **0**
Minimum: 0.0

value The target vaue for the distance check (Angstroms)

Valid values: reals
Default value: **0**
Minimum: 0.0

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩
The numbers of two atoms which are to be considered are to have their distance checked during a conformational search. Note that specifying a-b is the same as specifying b-a.

dynamics

Used to set options associated with MacroModel molecular or stochastic dynamics procedures.

Syntax:

dynamics *equilibration*=⟨ x ⟩ *method*=molecular | stochastic *sample*=⟨ n ⟩ *shake*=nothing | hydrogens | all *simulation*=⟨ x ⟩ *temperature*=⟨ x ⟩ *timestep*=⟨ x ⟩

Options:

equilibration
The time used for equilibration for the equilibration part of the simulation in ps.

Valid values: reals
Default value: **1**
Minimum: 0.0

method This option determines which dynamics method will be used.

Valid values: molecular
stochastic
Default value: **stochastic**

sample The number of structures to be sampled during the simulation.

Valid values: integers
Default value: **0**
Minimum: 0

shake This option determines how the SHAKE algorithm will be applied during the simulation.

Valid values: nothing
hydrogens
all
Default value: **nothing**

simulation
The total time to be used in the simulation.

Valid values: reals
Default value: **10**
Minimum: 0.0

temperature
The temperature at which the simulation is to be run in Kelvin.

Valid values: reals
Default value: **300**
Minimum: 0.0

timestep The timesetp which is to be used for the simulation in fs.

Valid values: reals
Default value: **1.5**
Minimum: 0.0

ecalc

This is a standard alias for **calcenergy** (see [calcenergy], page 44).

ejob

This keyword is used to set various options associated with running MacroModel jobs.

Syntax:

ejob *host*=⟨ text ⟩ *incorporate*=append | replace | ignore
job=⟨ text ⟩ *login*=⟨ text ⟩ *structure_source*=selected_entries |
workspace | file

Options:

host The name of the host for the MacroModel job.

Valid values: text strings
Default value:

incorporate
How the results are to be incorporated into the project. This can be done with replacement of the existing entries or by appending as new entries to the project or by ignoring the final results.

Valid values: append
replace
ignore
Default value: **append**

job The name for the MacroModel job.

Valid values: text strings
Default value: **mmodtmp**

login The login name under which a MacroModel will be run.

Valid values: text strings
Default value:

structure_source
Whether to use the selected entries in the current project or what is in the workspace as input for the job.

Valid values: selected_entries
workspace
file
Default value: **workspace**

elementlabels

Set the element label options.

Syntax:

elementlabels *labelscheme*=heterohydro | hetero | off

Options:

labelscheme

Three Options for displaying element labels: 1) on all non_carbon atoms, including hydrogen. 2) on all non-carbon atoms, except hydrogen (i.e., not on hydrogen) 3) off (no element labels displayed).

Valid values: heterohydro
hetero
off
Default value: **off**

embrace

Used to control the eBMrAcE feature of MacroModel.

Syntax:

embrace *energymode*=interaction | energydiff *inputfile*=⟨ text ⟩ *ligandsmode*=inputfile | selectedentries *ouput*=complexes | ligands | all *receptor*=⟨ text ⟩ *receptormode*=firstitem | entry

Options:

energymode

This option determines which energy mode will be used by eBMrAcE.

Valid values: interaction
energydiff
Default value: **energydiff**

inputfile The name of the eMBrAcE ligands input file.

Valid values: text strings
Default value:

ligandsmode

This option determines which source of ligands will be used by eBMrAcE.

Valid values: inputfile
selectedentries
Default value: **inputfile**

ouput This option determines what structural output will be produced.

Valid values: complexes
ligands
all
Default value: **complexes**

receptor This option specifies which entry in the project is to be treated as the receptor.

Valid values: text strings
Default value:

receptormode
This option determines which kind of receptor will be used by eBMrAcE.

Valid values: firstitem
entry
Default value: **firstitem**

embracecsearch

Defines settings for eMBrAcE conformational searching in MacroModel.

Syntax:

embracecsearch *max_distance_lmcs*=⟨ x ⟩ *maxdist*=⟨ x ⟩
method=embrace_mcmm | embrace_lmcs | embrace_mixed_lmcs
min_distance_lmcs=⟨ x ⟩ *numconfs*=⟨ n ⟩ *numsteps*=⟨ n ⟩
numstructures=⟨ n ⟩ *probability_tors_lmcs*=⟨ x ⟩ *window*=⟨ x ⟩

Options:

max_distance_lmcs
The maximum possible distance for the fastest moving atom in each LMCS move.

Valid values: reals
Default value: **6**
Minimum: 0.0

maxdist Maximum distance between atoms in equal structures.

Valid values: reals
Default value: **0.25**
Minimum: 0.0

method This determines which method will be used to perform the eMBrAcE conformational search.

Valid values: embrace_mcmm
embrace_lmcs
embrace_mixed_lmcs
Default value: **embrace_mcmm**

min_distance_lmcs
The minimum possible distance for the fastest moving atom in each LMCS move.

Valid values: reals
Default value: **3**
Minimum: 0.0

numconfs The number of input conformations available to seed each search.

Valid values: integers
Default value: **1**
Minimum: 0

numsteps An option which sets the number of steps which will be performed during the eMBrAcE conformational search.

Valid values: integers
Default value: **100**
Minimum: 0

numstructures
The number of structures that will be saved at the end of each search.

Valid values: integers
Default value: **1**
Minimum: 0

probability_tors_lmcs
The probability that a TORS/MOLS move will be made during an LMCS search.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

window The energy window (in kJ/mol) within which structures will be saved.

Valid values: reals
Default value: **500**
Minimum: 0.0

endundoblock

End the current undoable command block.

Syntax:

endundoblock

energykill

Specifies a running job to be killed. This will force the program running the job to be stopped.

Syntax:

energykill ⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩

Operands:
⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩
The ID or name of the job which is to be put to killed. This must be the ID or name of a job which is currently running or sleeping.
Aliases:
kill (see [kill], page 187)

energymonitor

Specifies a job to monitor.

Syntax:

energymonitor ⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩
⟨project_name⟩

Operands:
⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩
The ID or name of the job which is to be monitored. If no operand is specified then the job which was most recently monitored will be monitored.
Aliases:
monitor (see [monitor], page 211)

energysleep

Specifies a job to be put to sleep

Syntax:

energysleep ⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩

Operands:
⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩
The ID or name of the job which is to be put to sleep. This must be the ID or name of a job which is currently running.
Aliases:
sleep (see [sleep], page 390)

energystart

Start a MacroModel job with the current energy settings.

Syntax:

energystart *onscreen*=yes | no ⟨job_name⟩

Options:

onscreen Determines whether the on-screen structure will be written and used as the .dat file for the MacroModel job.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨job_name⟩
The name of the job which is to be started. This operand is not the name of a file and should not contain any suffix. From this name the files "job_name".com and "job_name".dat will be created.

energystop

Specifies a running job to be stopped. The program running the job will be requested to stop but may take some time to respond to the command.

Syntax:

energystop ⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩

Operands:
⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩
The ID or name of the job which is to be stopped. This must be the ID or name of a job which is currently running or sleeping.
Aliases:
stop (see [stop], page 391)

energytask

Determines which energy task is currently being set up.

Syntax:

energytask ecalc | mini | moldyn | drive | mult | csearch | mcsd | minta | embrace | embracecsearch

Operands:
ecalc | mini | moldyn | drive | mult | csearch | mcsd | minta | embrace | embracecsearch
The type of energy task which is to be set up. The operand cannot be abreviated and must be given in full.

energyupdate

Specifies a running job to be updated.

Syntax:

energyupdate ⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩

Operands:
⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩

The ID or name of the job which is to be put to updated. This is not the name of a file and no suffix is required. At present only MacroModel conformational searches will pay any attention to a request to update, however it is no error for an update to be issued for any type of job.
Aliases:
update (see [update], page 420)

energyupdatejobstatus

update the job status of all the jobs in the monitor panel.

Syntax:

energyupdatejobstatus

energywake

Specifies a sleeping job to be woken up.

Syntax:

energywake ⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩

Operands:
⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩
The ID of the job which is to be put to woken up. This must be the ID of a job which is currently sleeping.
Aliases:
wake (see [wake], page 427)

enhance3d

Fit clipping planes to displayed structure. If display stereo is on, this may help prevent conflicts between the clipping range and the stereo angle that lead to clipping of the displayed structure or flatter appearance of the structure in stereo, especially when zooming in. If display perspective is on, this may help prevent conflicts between the front clipping plane and the viewing

distance (which corresponds to the perspective scale) that lead to clipping of the displayed structure or reduced perspective scaling, especially when zooming in. If display fog is on, this helps maximize the shading contrast between near and far atoms and helps make the front part of the displayed structure more visible.

Syntax:

enhance3d

entrycombine

Combine selected project entries into a single entry.

Syntax:

entrycombine *replace*=yes | no ⟨new_name⟩ ⟨ESL⟩

Options:

replace This determines whether the combined entry is allowed to replace an existing entry having the name <new_name>. If the name of the combined entry matches the name of an entry in the workspace, the entry in the workspace is also replaced. If the replace option is off, the user will be asked to accept a unique name based upon <new_name>.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨new_name⟩ ⟨ESL⟩
The name to be given to the single combined project entry, followed by a valid ESL expression. If no name is specified, no rename will be done. The ESL expression specifies which entries are to be combined.

entrycopyprop

Set property values for selected project entries from another property. The to property value will be copied from the from property for the entries selected by the ESL expression. If the to property does not already exist, it will

be created. When the data type for the two properties is not the same, a reasonable attempt is made to convert between them. If a selected entry has no value for the from property, or the conversion from a string value fails, the value is generally cleared for the to value for that entry. Entry names are never cleared.

Syntax:

entrycopyprop *from*=⟨text⟩ *to*=⟨text⟩ ⟨ESL⟩

Options:

from The name of the property which provides the values to be copied. This can be either the user name or the m2io data name for the property.

Valid values: text strings
Default value:

to The name of the property to be modified. This can be either the user name or the m2io data name for the property, if the to property exists. If the to property does not exist, it will be created using the data type of the from property, with user as the author. If the property is the entry name (e.g. Entry Name or s_m_entry_name), an entryrename with replace=no will be done. If the property is the 'included in Workspace' property (In or b_m_entry_in_workspace), then entrywsinclude or entrywsexclude will be done, if needed.

Valid values: text strings
Default value:

Operands:
⟨ESL⟩
The ESL expression specifies for which entries the property values are to be copied.

entrydelete

Delete each selected project entry.

Syntax:

entrydelete ⟨ESL⟩

Operands:
⟨ESL⟩
A valid ESL specification. Delete those entries which match the ESL description.

entrydisassociate

Create a new entry for each molecule in each selected project entry. Each of these new entries will be given a unique name, based upon the name of its originating entry. Any of the originating entries which are in the workspace are replaced by their disassociated molecule entries.

Syntax:

entrydisassociate ⟨ESL⟩

Operands:
⟨ESL⟩
A valid ESL specification. Disassociate those entries which match the ESL description.

entrydragselection

Move current entry selection to the specified table row.

Syntax:

entrydragselection ⟨row⟩ ⟨table⟩

Operands:
⟨row⟩ ⟨table⟩
The destination row number. In place of the row number, "top" may be used to specify the first row of the table (row 1) and "bottom" may be used to specify placement after the last unselected row (last_row + 1). The selected entries will be moved as a block, placing the first selected entry below the first unselected entry which is above the destination row. If there is no unselected entry above the destination row, the selected entries will be

moved to the top of the table. The name of the table to use as the source for the row number. If the table operand is missing, the current or default table (1) will be used.

entryduplicate

Create a new duplicate entry for each selected project entry. Each of these new entries will be given a unique name, based upon the name of its originating entry.

Syntax:

entryduplicate ⟨ESL⟩

Operands:

⟨ESL⟩

A valid ESL specification. Duplicate those entries which match the ESL description.

entryexport

Export selected entries from the current project to a file or a number of files. If the export command is issued with options, but without a filename then nothing is written but the options are updated. An export command with both options and a file name specified will result in the export being performed with the new options to the specified file(s). When exporting each entry to an individual file, the given file name will be the base file name which is used to name those individual files by adding their associated postfixes to the base file name.

Syntax:

entryexport *append*=yes | no *files*=single | individual *format*=mmod | pdb | mol2 | maestro | sd | jaguarinput | gaussinput | biograf | xyz | jaguaroutput | gaussian92 | gaussian94 | gamess | mopaccartesian | mopacinternal | mopacoutput | babelpdb | mdl | babelmol | babelmol2 | spartan | spartansemi | spartanmm | gamessinput | gausscartesian | gaussianz | jaguarzmatrix | jaguarcartesian | any *graphical*=yes | no *names*=withentry | withnumber | entry *source*=workspace | selected [⟨filename⟩]

Options:

append This option determines whether to append to the file which is going to be written. It will be grayed out when PDB format is selected.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

files This option sets the number of export files. The two options are: export entries to a single file (0 default) or export each entry to an individual file.

Valid values: single
individual
Default value: **single**

format This option sets the format of the file to be written. Valid values are "mmod", "pdb", "mol2", "maestro" or "sd".

Valid values: mmod
pdb
mol2
maestro
sd
jaguarinput
gaussinput
biograf
xyz
jaguaroutput
gaussian92
gaussian94
gamess
mopaccartesian
mopacinternal
mopacoutput
babelpdb
mdl
babelmol
babelmol2
spartan
spartansemi
spartanmm
gamessinput
gausscartesian
gaussianz
jaguarzmatrix
jaguarcartesian
any
Default value: **maestro**

graphical This option determines whether to export graphical information when an entry is being saved to disk.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

names This option sets output file names when a number of files are exported. There are three options: file name + entry name, file name + number, just entry name. This option will be grayed out if only one file is exported.

Valid values: withentry
withnumber
entry
Default value: **withentry**

source Whether to export selected entries in the current project table or to export what is in the workspace, including scratch entry.

Valid values: workspace
selected
Default value: **selected**

Operands:

[⟨filename⟩]

The name of the file to which entries will be written. If no name is specified, then no export will be done.

entryexportspreadsheet

Export selected entries from the current project to a file for use in a spreadsheet. This can be comma-separated value (.csv) format or tab-delimited format.

Syntax:

entryexportspreadsheet *columns*=all | subset *delimiter*=⟨text⟩ *rows*=all | subset | selected | included ⟨filename⟩

Options:

columns This option sets the columns to be output. The columns can either be all or subset .

Valid values: all
subset
Default value: **all**

delimiter This option sets the delimiter to use to separate columns.

Valid values: text strings
Default value: **,**

rows This option sets the rows to be written out. The rows can be all , subset , selected , or included .

Valid values: all
subset
selected
included
Default value: **all**

Operands:

⟨ filename ⟩

The name of the file to which entries will be written. If no name is specified, then no export will be done.

entryextendselection

Extend current entry selection to encompass the specified entry.

Syntax:

entryextendselection ⟨ entry_name ⟩

Operands:

⟨ entry_name ⟩

The name of an entry in the current project. A range selection will be done between currently selected entries and the specified entry.

entryextendwsinclude

Extends the included entries to encompass the selected project entry.

Syntax:

entryextendwsinclude ⟨ ESL ⟩

Operands:

⟨ ESL ⟩

A valid ESL specification. Extends included entries to encompass the entries which match the ESL description into the workspace.

entryimport

Import structures into the current project.

Syntax:

entryimport *all*=yes | no *allfiles*=yes | no *end*=yes | no *format*=mmod | pdb | mol2 | maestro | sd | jaguarinput | gaussinput | biograf | xyz | jaguaroutput | gaussian92 | gaussian94 | gamess | mopaccartesian | mopacinternal | mopacoutput | babelpdb | mdl | babelmol | babelmol2 | spartan | spartansemi | spartanmm | gamessinput | gausscartesian | gaussianz | jaguarzmatrix | jaguarcartesian | any *graphical*=yes | no *start*=⟨n⟩ *total*=⟨n⟩ *wsinclude*=none | first | all *wsreplace*=yes | no [⟨filename⟩]

Options:

all This determines whether all structures will be imported, or just a specified range.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

allfiles This determines whether all files in the directory given in the operand will be imported.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

end This determines if all structures in the file are to be imported starting from the structure specified by start.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

format This option sets the format of the file to be read. Valid values are “maestro”, “mmod”, “pdb”, “mol2”, or “sd”.

Valid values: mmod
pdb
mol2
maestro
sd
jaguarinput
gaussinput
biograf
xyz
jaguaroutput
gaussian92
gaussian94
gamess
mopaccartesian
mopacinternal
mopacoutput
babelpdb
mdl
babelmol
babelmol2
spartan
spartansemi
spartanmm
gamessinput
gausscartesian
gaussianz
jaguarzmatrix
jaguarcartesian
any

Default value: **maestro**

graphical This determines whether the file will be read with just the geometric information or if the visual information in the file will be read and used for displaying

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

start This option sets the number of the first structure to be imported, if not importing all.

Valid values: integers
Default value: **1**
Minimum: 1

total The total number of structures to be imported from the file, if not importing all structures.

Valid values: integers

Default value: **1**
Minimum: 1

wsinclude This option determines which of the imported structures are to be included in the workspace. Valid values are "none", "first", or "all".

Valid values: none
first
all
Default value: **first**

wsreplace This determines whether the structures currently in the workspace will be replaced by the included imported structures, or whether they will be be included in the workspace also.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
[⟨filename⟩]
The name of the file from which structures will be imported. If no name is specified, then no import will be done.

entryimportspreadsheet

Import entries from a file into the project table.

Syntax:

entryimportspreadsheet *delimiter*=comma | tab | userdefined
userdelimiter=⟨text⟩ ⟨filename⟩ ⟨import_key⟩ ⟨proj_key⟩

Options:

delimiter This option sets the delimiter to use to separate columns.

Valid values: comma
tab
userdefined
Default value: **comma**

userdelimiter
This option sets the delimiter defined by the user to separate columns.

Valid values: text strings
Default value:

Operands:
⟨ filename ⟩ ⟨ import_key ⟩ ⟨ proj_key ⟩
The name of the file from which entries will be read. The key to be considered as the reference from file. Property in project that maps to import_key.

entryimportvibration

Import vibration data for the given entry

Syntax:

entryimportvibration *entry*=⟨ text ⟩ ⟨ filename ⟩

Options:

entry This is the entry which the vibration data will be attached to.
Valid values: text strings
Default value:

Operands:
⟨ filename ⟩
The name of the file from which vibration data will be imported.

entryinvertselection

Invert the selection state of all project entries.

Syntax:

entryinvertselection

entryrename

Rename selected project entries.

Syntax:

entryrename *replace*=yes | no ⟨new_name⟩ ⟨ESL⟩

Options:

replace This determines whether a single selected entry is allowed to replace an existing entry having the name <new_name>. Otherwise, the user will be asked to accept a unique name based upon <new_name>.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨new_name⟩ ⟨ESL⟩
The new name to be given to a single selected project entry, or the basename for multiple selected project entries from which unique names will be derived for each. The new name should be followed by a valid ESL expression to specify which entries are to be renamed. If no name is specified, no rename will be done.

entryselect

Select specified entries in current project.

Syntax:

entryselect ⟨ESL⟩

Operands:
⟨ESL⟩
A valid ESL specification. Selects those entries which match the ESL description, adding to the currently selected entries.

entryselectall

Select all project entries.

Syntax:

entryselectall

entryselectonly

Select only the specified entries in current project.

Syntax:

entryselectonly ⟨ESL⟩

Operands:

⟨ESL⟩

A valid ESL specification. Selects only those entries which match the ESL description, unselecting all other entries.

entryselectonlyrow

Select only the entry corresponding to the given row in the current project.

Syntax:

entryselectonlyrow ⟨row⟩

Operands:

⟨row⟩

A row number between 1 and the total number of rows. Selects only the entry corresponding to the given row, unselecting all other entries.

entryselectrandom

Selects a random set of entries.

Syntax:

entryselectrandom *entries*=selected | all *percentage*=⟨ n ⟩ None

Options:

entries This determines whether the random entries will be chosen from the selected entries or from all entries.

Valid values:	selected all
Default value:	**selected**

percentage This is the percentage of entries to select.

Valid values:	integers
Default value:	**50**
Minimum:	0
Maximum:	100

Operands:
None
None

entryselectrow

Select the entry corresponding to the given row in the current project.

Syntax:

entryselectrow ⟨ row ⟩

Operands:
⟨ row ⟩
A row number between 1 and the total number of rows. Selects the entry corresponding to the given row.

entrysetprop

Set property value for selected project entries.

Syntax:

entrysetprop *property*=⟨text⟩ *value*=⟨text⟩ ⟨ESL⟩

Options:

property The name of the property to be modified. This can be either the user name or the m2io data name for the property. If the property is the entry name (e.g. Entry Name or s_m_entry_name), an entryrename with replace=no will be done. If the property is the 'included in Workspace' property (In or b_m_entry_in_workspace), then entrywsinclude or entrywsexclude will be done, if needed.

Valid values: text strings
Default value:

value The value to be set for the selected entries. For Boolean properties, legal values are yes, no, true, false, on, off, and 1, 0.

Valid values: text strings
Default value:

Operands:
⟨ESL⟩
The specified property value will be assigned to the entries selected by the ESL expression.

entrysettitle

Set entry title.

Syntax:

entrysettitle ⟨title⟩ ⟨ESL⟩

Operands:
⟨title⟩ ⟨ESL⟩
New entry title and a valid ESL specification. Set title for those entries which match the ESL description from the workspace.

entryshowall

Creates a entry subset in the project table consisting of all the entries in project.

Syntax:

entryshowall

entrytable

This keyword is used to set various options associated with the project entry table (Project Table).

Syntax:

entrytable *swapnametitle*=yes | no

Options:

swapnametitle
: This determines whether the table is shown and otherwise treated as if the entry name and title columns are swapped with respect to their positions in the project table.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

entryunselect

Unselect specified entries in current project.

Syntax:

entryunselect ⟨ESL⟩

Operands:
⟨ESL⟩
A valid ESL specification. Unselect those entries which match the ESL description.

entryunselectall

Unselect all project entries.

Syntax:

entryunselectall

entryunselectrow

Unselect the entry corresponding to the given row.

Syntax:

entryunselectrow ⟨row⟩

Operands:
⟨row⟩
A row number between 1 and the total number of rows in the project. Unselects the entry corresponding to the given row number.

entrywscreate

Create project entry from atoms in the workspace.

Syntax:

entrywscreate *replace*=yes | no ⟨new_name⟩ ⟨ASL⟩

Options:

replace This determines whether the created entry is allowed to replace an existing entry having the name <new_name>. Otherwise, the user will be asked to accept a unique name based upon <new_name>.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
⟨new_name⟩ ⟨ASL⟩
The name to be given to the created project entry, followed by a valid ASL expression to specify which atoms constitute the new project entry. If no name is specified, no operation will be done.

entrywsexclude

Exclude selected project entries from the workspace.

Syntax:

entrywsexclude ⟨ESL⟩

Operands:
⟨ESL⟩
A valid ESL specification. Excludes those entries which match the ESL description from the workspace.

entrywsexcludenotfixed

Exclude all unfixed entries from the workspace.

Syntax:

entrywsexcludenotfixed

entrywsinclude

Include selected project entries into the workspace.

Syntax:

entrywsinclude ⟨ESL⟩

Operands:
⟨ESL⟩
A valid ESL specification. Include those entries which match the ESL description into the workspace.

entrywsincludelock

Locks the given entries into the workspace.

Syntax:

entrywsincludelock ⟨ESL⟩

Operands:
⟨ESL⟩
A valid ESL specification. Locks the given entries into the workspace.

entrywsincludeonly

Include only selected project entries in the workspace. Exclude all others.

Syntax:

entrywsincludeonly ⟨ESL⟩

Operands:
⟨ESL⟩
A valid ESL specification. Include only those entries which match the ESL description in the workspace. Exclude all others.

eplayergoto

Go to the specified entry in the ordered sequence of selected project entries and pause there. The table specified in the eplayersettings command determines the sequence order.

Syntax:

eplayergoto ⟨entry_name⟩

Operands:
⟨entry_name⟩
The name of the entry, within the ordered sequence of selected project entries, which is to be included in the workspace. This frame is recorded as an option to the eplayersettings command.

eplayergotofirst

Go to the first entry in the ordered sequence of selected project entries. The table specified in the eplayersettings command determines the sequence order.

Syntax:

eplayergotofirst

eplayergotolast

Go to the last entry in the ordered sequence of selected project entries. The table specified in the eplayersettings command determines the sequence order.

Syntax:

eplayergotolast

eplayersettings

Set eplayer state variables.

Syntax:

eplayersettings *frame*=⟨ text ⟩ *frameduration*=⟨ x ⟩
playmode=loop | reverse | once *playsync*=yes | no
referentry=⟨ text ⟩ *script*=noaction | current | file
scriptfile=⟨ text ⟩ *superimpose*=none | previous | reference
table=⟨ text ⟩ *title*=⟨ text ⟩

Options:

frame This option sets the name of the entry being displayed.
Valid values: text strings
Default value:

frameduration
This option determines the minimum duration, in seconds, of each displayed (entry) frame during continuous play. The actual

frame duration may be longer than the specified value, due to time required for drawing and screen update.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 5.0

playmode This option sets the mode for continuous play. Valid values are "loop", "reverse", or "once". These cause play to continue, change direction, or stop, respectively, when reaching either end of the current entry selection.

Valid values: loop
reverse
once
Default value: **once**

playsync If true, save Workspace changes during continuous play.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

referentry This option sets the name of the reference entry used when the superimpose option is set to "reference". The reference entry must exist in the current project in order to use this option.

Valid values: text strings
Default value:

script This option sets the action for each step in the ePlayer. Valid values are "noaction", "current", or "file". At each step, ePlayer may take no action, run current command script, or run command script from a script file.

Valid values: noaction
current
file
Default value: **noaction**

scriptfile This option sets the name of the script file used when the script option is set to "Execute Command Script From File".

Valid values: text strings
Default value:

superimpose
This option sets the type of superposition done for entries displayed by the eplayer. Valid values are "none", "previous", or "reference". These cause the incoming entry to be displayed with its current coordinates, with its atoms superimposed upon those of the outgoing entry, or with its atoms superimposed upon those of the reference structure for registration.

	Valid values:	none previous reference
	Default value:	**none**
table	This option identifies the project table which is used to define to sequence order of the selected entries in the project.	
	Valid values:	text strings
	Default value:	
title	This option sets the title of the entry being displayed.	
	Valid values:	text strings
	Default value:	

eplayerstepahead

Go to the next entry in the ordered sequence of selected project entries, if there is one after the frame specified in the eplayersettings command. The table specified in the eplayersettings command determines the sequence order.

Syntax:

eplayerstepahead

eplayerstepback

Go to the previous entry in the ordered sequence of selected project entries, if there is one before the frame specified in the eplayersettings command. The table specified in the eplayersettings command determines the sequence order.

Syntax:

eplayerstepback

errorcheck

Specifies optional error checking to be performed.

Syntax:

errorcheck *ct*=yes | no

Options:

ct An option which determines whether mmct error checking is enabled during certain (primarily graphical) operations.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

excludedvolumesmarkersettings

Set graphical data of Phase excluded volume markers.

Syntax:

excludedvolumesmarkersettings *ambient*=⟨ x ⟩ *diffuse*=⟨ x ⟩ *emission*=⟨ x ⟩ *radius*=⟨ x ⟩ *shininess*=⟨ x ⟩ *slices*=⟨ n ⟩ *specular*=⟨ x ⟩ *stacks*=⟨ n ⟩ *transparency*=⟨ x ⟩

Options:

ambient Set material property - ambient, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.4**
Minimum: 0.0
Maximum: 1.0

diffuse Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.4**
Minimum: 0.0
Maximum: 1.0

emission Set material property - emission, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

radius The radius of QSAR markers.

Valid values: reals
Default value: **1**
Minimum: 0.1

shininess Set material property - shininess, for front face.

Valid values: reals
Default value: **80**
Minimum: 0.0
Maximum: 128.0

slices Set the slices of drawing sphere.

Valid values: integers
Default value: **18**
Minimum: 2

specular Set material property - specular, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

stacks Set the stacks of drawing sphere.

Valid values: integers
Default value: **9**
Minimum: 2

transparency
The transparency of QSAR markers.

Valid values: reals
Default value: **20**
Minimum: 0.0
Maximum: 100.0

extenddisplaytopose

This command displays the specified structure from the pose set and any poses between it and the nearest displayed poses before and after it in the list. No poses will be undisplayed (i.e. removed from the workspace).

Syntax:

extenddisplaytopose ⟨pose_name⟩

Operands:

⟨pose_name⟩

The name of the pose structure.

fileread

Read a structure file. If a filename is given then structures will be read from that file. If an explicit "start=" option is not included then starting structure is incremented at each read.
For example:
fileread start=1 fileread mmodtmp.dat fileread start=1 test.dat

Syntax:

fileread *delete*=yes | no *format*=mmod | pdb | mol2 | maestro | sd | jaguarinput | gaussinput | biograf | xyz | jaguaroutput | gaussian92 | gaussian94 | gamess | mopaccartesian | mopacinternal | mopacoutput | babelpdb | mdl | babelmol | babelmol2 | spartan | spartansemi | spartanmm | gamessinput | gausscartesian | gaussianz | jaguarzmatrix | jaguarcartesian | any *ginfo*=yes | no *start*=⟨n⟩ *tile*=yes | no *total*=⟨n⟩ [⟨file_name⟩]

Options:

delete This determines whether the currently displayed structure will be deleted before the new structure is read in.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

format This option sets the format of the file to be read. Valid values are "maestro", "mmod", "pdb", "mol2", or "sd".

Valid values: mmod
pdb
mol2
maestro
sd
jaguarinput
gaussinput
biograf
xyz
jaguaroutput
gaussian92
gaussian94
gamess
mopaccartesian
mopacinternal
mopacoutput
babelpdb
mdl
babelmol
babelmol2
spartan
spartansemi
spartanmm
gamessinput
gausscartesian
gaussianz
jaguarzmatrix
jaguarcartesian
any
Default value: **maestro**

ginfo This determines whether the file will be read with just the geometric information or if the visual information in the file will be read and used for displaying the structures.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

start This option sets the number of the first structure to be read

Valid values: integers
Default value: **1**
Minimum: 1

tile This determines whether the structures displayed on screen will be "tiled".

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

total The total number of structures to be read from the file

Valid values: integers
Default value: **1**
Minimum: 1

Operands:
[⟨file_name⟩]
The name of the file from which the structure will be read. If no name is specified, then no file read will be done.
Aliases:
read (see [read], page 360)

filewrite

If the write command is issued with options, but without a filename then nothing is written but the options are updated. If the write command alone is issued then writing is performed to the currently open file with the current options. A write command with both options and a file name specified will result in the write being performed with the new options to the specified file.

Syntax:

filewrite *append*=yes | no *displayed_atoms*=yes | no *format*=mmod | pdb | mol2 | maestro | sd | jaguarinput | gaussinput | biograf | xyz | jaguaroutput | gaussian92 | gaussian94 | gamess | mopaccartesian | mopacinternal | mopacoutput | babelpdb | mdl | babelmol | babelmol2 | spartan | spartansemi | spartanmm | gamessinput | gausscartesian | gaussianz | jaguarzmatrix | jaguarcartesian | any *graphical*=yes | no *separate*=yes | no *title*=⟨text⟩ ⟨file_name⟩

Options:

append This option determines whether to append to the file which is going to be written

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

displayed_atoms
This option determines whether the displayed atoms will be saved. This option is not currently supported.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

format This option sets the format of the file to be read. Valid values are "mmod", "pdb" or "mol2".

Valid values:
mmod
pdb
mol2
maestro
sd
jaguarinput
gaussinput
biograf
xyz
jaguaroutput
gaussian92
gaussian94
gamess
mopaccartesian
mopacinternal
mopacoutput
babelpdb
mdl
babelmol
babelmol2
spartan
spartansemi
spartanmm
gamessinput
gausscartesian
gaussianz
jaguarzmatrix
jaguarcartesian
any
Default value: **maestro**

graphical This option determines whether to write out graphical information when a structure is being saved to disk.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

separate This option determines whether to write out the CT as one structure or separate structures. This option is not currently supported.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

title The title for the output structure.

Valid values: text strings
Default value:

Operands:

⟨ file_name ⟩

The name of the file to which the structure will be written.

Aliases:

write (see [write], page 429)

filter

Creates a new entry filter. The filter name must be a single token (or "quoted" if multiple tokens). A filter can be redefined by specifying a new definition.

Syntax:

filter ⟨ filter_name ⟩ ⟨ ESL-definition ⟩

Operands:

⟨ filter_name ⟩ ⟨ ESL-definition ⟩

The name which will be applied to the filter. If the name contains embedded spaces then it must be enclosed in double quotation marks.

find

Find atom(s), residue(s), chain, or molecule described by the ASL operand.

Syntax:

find *atombyname*=⟨ text ⟩ *atombynum*=⟨ n ⟩ *byasl*=⟨ text ⟩ *center*=yes | no *chainname*=⟨ text ⟩ *findmethod*=bynumber | byname *findtype*=atom | residue | chain | molecule | asl *hidemarkers*=yes | no *inscode*=⟨ text ⟩ *label*=yes | no *markall*=yes | no *matchesnum*=⟨ text ⟩ *molnum*=⟨ n ⟩ *resnum*=⟨ n ⟩ ⟨ ASL ⟩

Options:

atombyname
This option determines name of the atom to be found.

Valid values: text strings
Default value:

atombynum
This option determines value of the atom number to be found

Valid values: integers
Default value: **1**
Minimum: 1

byasl This option determines the ASL specification fo the objects to be found (atom, residue, chain ot molecule).

Valid values: text strings
Default value:

center This option determines whether the structure will center on the found atom or on the centroid of the atom(s) [of residue, chain or molecule].

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

chainname
This option determines the chain name to be found, or the chain name in which to find a specified residue.

Valid values: text strings
Default value:

findmethod
This option determines the method used to specify an atom to be found, including By Number and By Name .

Valid values: bynumber
byname
Default value: **bynumber**

findtype This option determines the find type, including five options of atom, residue, chain, molecule and asl.

Valid values: atom
residue
chain
molecule
asl
Default value: **atom**

hidemarkers
This option determines whether all the markers should be hiden (TRUE) ot not (FALSE)
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

inscode This option determines value of the insertion code of the residue to be found.
Valid values: text strings
Default value:

label This option determines whether the found atoms will be labeld.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

markall This option determines whether all the found atom(s) or atom(s) of residue, chain or molecule are marked (TRUE), or only the principal atom(s) are marked (FALSE).
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

matchesnum
This option shows the number of matched atoms over the total number of atoms in the structure.
Valid values: text strings
Default value: /

molnum This option determines the number of the molecule to be found
Valid values: integers
Default value: **1**
Minimum: 1

resnum This option determines value of the residue number to be found.
Valid values: integers
Default value: **1**

Operands:
⟨ASL⟩
A string in the atom specification language which describes the set of atoms which are to be found.

fit

Fit structures to the workspace.

Syntax:

fit ⟨ASL⟩

Operands:
⟨ASL⟩
If present,we do fit to screen on ASL, if not present we do fit on all atoms.

forcefield

Used to display the force field in the force field viewer.

Syntax:

forcefield

forcefieldbend

A keyword which controls the display of the bend interactions in the FF viewer.

Syntax:

forcefieldbend *select*=⟨n⟩ *show*=all | high | medium | low
sortbyenergy=yes | no

Options:

select Set the interaction which is currently selected

Valid values:	integers
Default value:	**0**
Minimum:	0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low
Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldele

A keyword which controls the display of the electrostatic interactions in the FF viewer.

Syntax:

forcefieldele *select*=⟨ n ⟩ *show*=all | high | medium | low
sortbyenergy=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**
Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low
Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldgbsolv

A keyword which controls the display of the GBSOLV interactions in the FF viewer.

Syntax:

forcefieldgbsolv *select*=⟨ n ⟩ *show*=all | high | medium | low
sortbyenergy=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**
Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low
Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldimproper

A keyword which controls the display of the improper torsion interactions in the FF viewer.

Syntax:

forcefieldimproper *select*=⟨ n ⟩ *show*=all | high | medium | low
sortbyenergy=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**
Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low
Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldsasolv

A keyword which controls the display of the SASOLV interactions in the FF viewer.

Syntax:

forcefieldsasolv *select*=⟨ n ⟩ *show*=all | high | medium | low
sortbyenergy=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**
Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low
Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldstretch

A keyword which controls the display of the stretch interactions in the FF viewer.

Syntax:

forcefieldstretch *select*=⟨ n ⟩ *show*=all | high | medium | low
sortbyenergy=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**
Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low
Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldtorsion

A keyword which controls the display of the torsion interactions in the FF viewer.

Syntax:

forcefieldtorsion *select*=⟨ n ⟩ *show*=all | high | medium | low
sortbyenergy=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**
Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low
Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldvdw

A keyword which controls the display of the VDW interactions in the FF viewer.

Syntax:

forcefieldvdw *select*=⟨ n ⟩ *show*=all | high | medium | low
sortbyenergy=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**
Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low
Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldview

Used to specify a .mmo file which contains the interactions to be displayed in the FF viewer.

Syntax:

forcefieldview ⟨mmo_file_name⟩

Operands:
⟨mmo_file_name⟩
The name of the .mmo file which is to be opened and displayed. The full filename, including the .mmo suffix must be specified.

forcefieldwilson

A keyword which controls the display of the wilson interactions in the FF viewer.

Syntax:

forcefieldwilson *select*=⟨n⟩ *show*=all | high | medium | low
sortbyenergy=yes | no

Options:

select Set the interaction which is currently selected
Valid values: integers
Default value: **0**
Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low
Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

formalcharge

Increment, decrement or set the formal charge for atoms which match the ASL specification.

Syntax:

formalcharge increment|decrement|⟨formal_charge⟩ ⟨ASL⟩

Operands:

increment|decrement|⟨formal_charge⟩ ⟨ASL⟩

The first operand is either the word "increment", the word "decrement", or an integer value represening the formal charge to be used for all atoms which match the specification. "increment" and "decrement" change the atom's current formal charge by +1 or -1 respectively. The second operand is a valid ASL string which defines the set of atoms which are to have their atom names changed. An error will be issued for the first atom in the set which cannot have its formal charge set (usually because no suitable charged type exists) and no further atoms will have their charge changed.

fragment

This command sets the fragment mode and the current fragment used for growing, placing and replacing.

Syntax:

fragment ⟨fragment_mode⟩ [⟨current_fragment⟩]

Operands:
⟨fragment_mode⟩ [⟨current_fragment⟩]
The first operand is the name of the fragment mode, this must be a valid name from the file $SCHRODINGER/mmshare-vX.X/data/res/mmfrag.ini. The optional second argument is the name of a fragment within that mode which is to be made the current fragment.

fragmenttype

This command sets the fragment type for the current fragment mode.

Syntax:

fragmenttype ⟨fragment_type⟩

Operands:
⟨fragment_type⟩
The first operand is the a qualifier for the fragment which some modes (such as furanose and pyranose) have. For example alpha or beta for pyranose or furanose sugars. The fragment_type must be a valid type for the current fragment mode.

frozenatom

Specifies a single atom to be frozen at its current position in a MacroModel calculation. Other atoms in the molecule will continue to interact with this atom. The frozen atom however will not "feel" any other atoms in the molecule.

Syntax:

frozenatom ⟨atom_number⟩

Operands:
⟨atom_number⟩

The number of an atom which is to be treated as frozen during a MacroModel calculation.

frozenset

Specifies a set of atoms to be frozen at their current positions in a MacroModel calculation. Other atoms in the molecule will continue to interact with these atoms. The frozen atoms however will not "feel" any other atoms in the molecule.

Syntax:

frozenset ⟨ASL⟩

Operands:

⟨ASL⟩

A string in the atom specification language which describes the set of atoms which are to be treated as frozen in a MacroModel calculation.

fuse

Fuse atom list pairs

Syntax:

fuse

geometrycleanup

Cleans up the geometry

Syntax:

geometrycleanup

glideactivesiteres

Defines a residue to be included in the active site for a Glide grid calculation.

Syntax:

glideactivesiteres ⟨ chain ⟩:⟨ molnum ⟩:⟨ resnum ⟩:⟨ insertioncode ⟩

Operands:
⟨ chain ⟩:⟨ molnum ⟩:⟨ resnum ⟩:⟨ insertioncode ⟩
A colon separated list giving the PDB chain name, the residue number and insertion code for a residue to be included in the list of active site residues.

glideactivesiteset

Defines a set from atoms, from which all residues will be added as part of the active site for Glide calculations.

Syntax:

glideactivesiteset ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A valid ASL specification. All residues which have any atoms in this set will be added to the list of residues that define the active site.

glidecalcboxfromligand

Uses the information specified in the glideligandgrid command to calculate the grid box for a Glide calculation.

Syntax:

glidecalcboxfromligand

glideconstraintregion

Specifies a constraint region in the receptor for a Glide calculation.

Syntax:

glideconstraintregion ⟨region_name⟩ ⟨cell⟩

Operands:

⟨region_name⟩ ⟨cell⟩

The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glideconstraintregionactivecell

Set an active cell for the region.

Syntax:

glideconstraintregionactivecell ⟨region_name⟩ ⟨active_cell⟩

Operands:

⟨region_name⟩ ⟨active_cell⟩

The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glideconstraintregiongrow

Grow cells for a region.

Syntax:

glideconstraintregiongrow ⟨region_name⟩

Operands:
⟨region_name⟩
The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glideconstraintregioninvisible

Set a constraint region as invisible in workspace.

Syntax:

glideconstraintregioninvisible ⟨region_name⟩

Operands:
⟨region_name⟩
The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glideconstraintregionshrink

Shrink cells for a region.

Syntax:

glideconstraintregionshrink ⟨region_name⟩

Operands:
⟨region_name⟩
The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glideconstraintregionvisible

Set a constraint region as visible in workspace.

Syntax:

glideconstraintregionvisible ⟨region_name⟩

Operands:
⟨region_name⟩
The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glidedisplayreceptor

Displays the receptor in the Workspace, clearing any existing scratch entries.

Syntax:

glidedisplayreceptor

glidedockregionnumatoms

Specifies the minimum number of atoms permitted to occupy this constraint region.

Syntax:

glidedockregionnumatoms ⟨region_name⟩ ⟨num_atoms⟩

Operands:
⟨region_name⟩ ⟨num_atoms⟩
The name of the region and the minimum number of atoms permitted for this region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glidedockregionselect

Selects a hydrophobic constraint region to be used as a constraint in the docking job.

Syntax:

glidedockregionselect ⟨region_name⟩

Operands:

⟨region_name⟩

The name of the region to be selected as a constraint in the docking job. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glidedockregionunselect

Unselects a hydrophobic constraint region so it will not be used as a constraint in the docking job.

Syntax:

glidedockregionunselect ⟨region_name⟩

Operands:

⟨region_name⟩

The name of the region to be unselected as a constraint in the docking job. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glidegridhydrophobic

Used to set all the options associated with the hydrophobic constraints in Glide grid generation.

Syntax:

glidegridhydrophobic *num_vertices*=⟨n⟩ *threshold*=⟨x⟩

Options:

num_vertices

At least this number of cell vertices must meet the threshold in order for the cell to be displayed.

Valid values: integers
Default value: **5**
Minimum: 1
Maximum: 8

threshold This is cutoff for which grid cells sre considered hydrophobic.

Valid values: reals
Default value: **-0.5**
Minimum: -0.65
Maximum: -0.01

glideligand

Specifies settings about the ligand(s) to be used in a Glide job.

Syntax:

glideligand *amidebondrotations*=yes | no *baddist*=⟨ x ⟩ *conformations*=usesupplied | generate | inplace *definereference*=yes | no *dockdisplayed*=yes | no *dockfromfile*=yes | no *dockrange*=all | specified *econfcut*=⟨ x ⟩ *endlig*=⟨ n ⟩ *format*=maestro | sd | mol2 | pdb *lig_ccut*=⟨ x ⟩ *lig_vscale*=⟨ x ⟩ *ligandsfile*=⟨ text ⟩ *ligandsource*=neither | entries | extfile *reflig*=yes | no *ringconf*=yes | no *startlig*=⟨ n ⟩

Options:

amidebondrotations
An option which allows amide bond rotations.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

baddist The distance criterion for bad internal contacts

Valid values: reals
Default value: **2.45**
Minimum: 0.0

conformations
Whether to use only supplied conformations, to generate them with confgen or to treat them "in-place".

Valid values: usesupplied
generate
inplace

Default value: **generate**

definereference
An option which determines if a reference ligand is to be defined.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

dockdisplayed
An option which determines if the currently displayed ligand will be docked.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

dockfromfile
[NOTE: This option is no longer used.] An option which determines ligands from external files are to be docked:

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

dockrange The range of structures in the file to be docked - whether to dock all from the file or a specified range.

Valid values: all
specified
Default value: **all**

econfcut The energy window for keeping conformations.

Valid values: reals
Default value: **12**
Minimum: 0.0

endlig The final structure from the file to be docked.

Valid values: integers
Default value: **1000**
Minimum: 0

format The format of the ligands.

Valid values: maestro
sd
mol2
pdb
Default value: **maestro**

lig_ccut The partial atomic charge below which ligand atoms are considered to be non-polar and will have their VDW radii scaled.

Valid values: reals
Default value: **0.15**
Minimum: 0.00000001

lig_vscale The scaling factor for the VDW radii of non-polar ligand atoms.

Valid values: reals
Default value: **0.8**
Minimum: 0.00000001

ligandsfile The file containing one or more ligands.

Valid values: text strings
Default value:

ligandsource
The source of the ligands to be docked

Valid values: neither
entries
extfile
Default value: **extfile**

reflig An option which determines if a reference ligand is to be used in this Glide job.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ringconf An option which allows ring flips

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

startlig The first structure from the file to be docked.

Valid values: integers
Default value: **1**
Minimum: 1

glideligandfile

Specifies a file containing one or more ligands to be used for a Glide docking run.

Syntax:

glideligandfile ⟨ligand_file_name⟩

Operands:
⟨ligand_file_name⟩
The name of a file containing one or more ligands.

glideligandgrid

Defines the ligand to be used to calculate the grid for the glide calculation.

Syntax:

glideligandgrid *ligandentry*=⟨text⟩ *ligandformat*=maestro | sd | mol2 | pdb *ligandindex*=⟨n⟩ *ligandsource*=displayed | entryname | extfile | none *usedisplayed*=yes | no ⟨ligand file name⟩

Options:

ligandentry

The name of the entry that will be used to define the grid for the Glide calculation.

Valid values: text strings
Default value:

ligandformat

The format of the file that will be used to calculate the grid box.

Valid values: maestro
sd
mol2
pdb
Default value: **maestro**

ligandindex

The index of the structure that is to be read as a ligand.

Valid values: integers
Default value: **1**
Minimum: 1

ligandsource

The source of the ligand to be used to define the Grid for the Glide calculation.

Valid values: displayed
entryname
extfile
none
Default value: **displayed**

usedisplayed

[NOTE: This option is no longer used.] An option which determines if the currently displayed ligand will be used to calculate the grid for the Glide calculation

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ligand file name⟩
The name of the file containing the ligand to be used to define the grid.

glidelocatehydrophobiccells

Runs sitemap and displays hydrophobic cells.

Syntax:

glidelocatehydrophobiccells

glideoutput

Controls a number of options for how the poses are output during a Glide calculation.

Syntax:

glideoutput *delpose*=⟨x⟩ *maxperlig*=⟨n⟩ *nreport*=⟨n⟩
outputdisp=poseviewer | ligandsonly *posedist*=⟨x⟩

Options:

delpose The maximum displacement for discarding duplicate poses.
Valid values: reals
Default value: **1.3**
Minimum: 0.0

maxperlig The maximum number of poses per ligand to be written.
Valid values: integers
Default value: **1**
Minimum: 0

nreport The number of coordinate sets to write to disk.
Valid values: integers
Default value: **1000**
Minimum: 1

outputdisp Whether the output structure file will be intended to be viewed with the Maestro pose viewer (and include the receptor structure) or is just to be the docked ligands.

Valid values: poseviewer
ligandsonly
Default value: **poseviewer**

posedist The distance criterion for distinct poses.

Valid values: reals
Default value: **0.5**
Minimum: 0.0

glidereceptorligand

Defines a on-screen molecule to be treated as the ligand for a Glide calculation

Syntax:

glidereceptorligand ⟨molecule_number⟩

Operands:
⟨molecule_number⟩
The number of a molecule to be included as the ligand.

glidereferenceligand

Specifies a file containing the reference ligand for a glide docking run.

Syntax:

glidereferenceligand *index*=⟨n⟩ *ligandentry*=⟨text⟩
ligandfile=⟨text⟩ *ligandformat*=maestro | sd | mol2 | pdb
ligandsource=displayed | entryname | extfile | none
use_displayed=yes | no

Options:

index The index of the reference ligand in the reference ligand file.

Valid values: integers

Default value: **1**
Minimum: 1

ligandentry
The name of the entry that will be used as the reference ligand.

Valid values: text strings
Default value:

ligandfile The name of the file containing the ligand structure that will be used as the reference ligand.

Valid values: text strings
Default value:

ligandformat
The format of the file that will be used to calculate the grid box.

Valid values: maestro
sd
mol2
pdb
Default value: **maestro**

ligandsource
The source of the ligand to be used as the reference ligand.

Valid values: displayed
entryname
extfile
none
Default value: **none**

use_displayed
[NOTE: This option is no longer used.] An option which determines if the reference ligand is that which is already displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

glidescoring

Controls a number of options for how the poses are scored.

Syntax:

glidescoring *cvcutoff*=⟨ x ⟩ *dielectric*=⟨ x ⟩ *hbcutoff*=⟨ x ⟩
itmax=⟨ n ⟩ *maxkeep*=⟨ n ⟩ *maxref*=⟨ n ⟩ *mlcutoff*=⟨ x ⟩
posedist=⟨ x ⟩ *scorecut*=⟨ x ⟩ *short_distance*=anneal | softcore

Options:

cvcutoff Reject poses if the Coulomb/VDW energy exceeds this value

Valid values: reals
Default value: **0**

dielectric The dielectric constant used in the grid energy calculation.

Valid values: reals
Default value: **2**
Minimum: 0.9999999999

hbcutoff Reject poses if the H-bond energy exceeds this value

Valid values: reals
Default value: **0**
Maximum: 0.00000001

itmax The maximum number of conjugate gradient steps

Valid values: integers
Default value: **100**
Minimum: 0

maxkeep The number of ligand poses on the coarse grid.

Valid values: integers
Default value: **5000**
Minimum: 1

maxref The maximum number of poses to keep after rough score refinement.

Valid values: integers
Default value: **400**
Minimum: 1

mlcutoff Reject poses if the metal-ligand interaction score exceeds this value.

Valid values: reals
Default value: **0**
Maximum: 0.00000001

posedist The distance criterion for distinct poses.

Valid values: reals
Default value: **0.5**
Minimum: 0.0

scorecut The window of rough scores for keeping poses.

Valid values: reals
Default value: **100**
Minimum: 0.0

short_distance
The short distance behavior of the potentials.

Valid values: anneal
softcore
Default value: **anneal**

glidesettings

Allows the settings of some values that determine how the overall Glide job runs.

Syntax:

glidesettings *function*=run | setup *griddirectory*=⟨text⟩ *gridfilename*=⟨text⟩ *maxatom*=⟨n⟩ *maxrotbonds*=⟨n⟩ *mode*=normal | accurate *numjobs*=⟨n⟩ *numprocs*=⟨n⟩ *receptor*=alone | withligand | ligandonly *recyclesteps*=⟨n⟩ *source*=structure | file *splitligands*=yes | no *writegrid*=yes | no

Options:

function Determines whether the Glide job is actually to dock some ligands or just calculate a grid for a receptor.

Valid values: run
setup
Default value: **setup**

griddirectory
The directory where the grid files are located.

Valid values: text strings
Default value:

gridfilename
The base name for the file which the receptor grid is to be written to or read from.

Valid values: text strings
Default value:

maxatom Any ligands in the input with more than this number of atoms will be skipped.

Valid values: integers
Default value: **80**
Minimum: 1
Maximum: 200

maxrotbonds
Any ligands in the input with more than this number of rotatable bonds will be skipped.

Valid values: integers
Default value: **15**
Minimum: 1

mode Determines how a number of other values are set to ensure the job runs either faster than normal or more accurate than normal.

Valid values: normal
accurate
Default value: **normal**

numjobs Number of jobs to split this docking job into

Valid values: integers
Default value: **1**
Minimum: 1

numprocs Number of processors to use for each job.

Valid values: integers
Default value: **1**
Minimum: 1

receptor Determines whether the structure displayed is just the receptor or a receptor + ligand.

Valid values: alone
withligand
ligandonly
Default value: **alone**

recyclesteps
Specifies the number of ligand recycling iterations in high accuracy docking jobs (glidesettings mode=accurate).

Valid values: integers
Default value: **5**
Minimum: 1

source Where the grid files come from: calculated from the structure or from previously calculated files.

Valid values: structure
file
Default value: **structure**

splitligands
Whether to split the ligand structures into separate files per job
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

writegrid An option which determines if the grid will be written following the job.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

glidesite

Settings for the site in a glide job.

Syntax:

glidesite *boxcenter*=ligand | residues | supplied *boxsize*=⟨ n ⟩ *display_center_box*=yes | no *display_enclosing_box*=yes | no *enclosingbox*=fitlargest | fitdisplayed | fitlength *maxliglength*=⟨ x ⟩ *recep_ccut*=⟨ x ⟩ *recep_vscale*=⟨ x ⟩ *xcent*=⟨ x ⟩ *ycent*=⟨ x ⟩ *zcent*=⟨ x ⟩

Options:

boxcenter Specifies how the box center is defined - by specifiying residues in the active site, by specifying a ligand or by using the values as input by the user.
Valid values: ligand
residues
supplied
Default value: **ligand**

boxsize The size of the enclosing box.
Valid values: integers
Default value: **10**
Minimum: 6
Maximum: 40

display_center_box
An option which determines if the box which contains the ligand center of motion will be displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

display_enclosing_box
An option which determines if the box which encloses the ligand center of motion box will be displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

enclosingbox
Specifies how the enclosing box is sized - based on the largest ligand, the displayed ligand or ligand length

Valid values: fitlargest
fitdisplayed
fitlength
Default value: **fitdisplayed**

maxliglength
The maximum ligand length expected.

Valid values: reals
Default value: **20**
Minimum: 0.0000000001

recep_ccut The partial atomic charge below which receptor atoms are considered to be non-polar and will have their VDW radii scaled.

Valid values: reals
Default value: **0.25**
Minimum: 0.00000001

recep_vscale
The scaling factor for the VDW radii of non-polar receptor atoms.

Valid values: reals
Default value: **1**
Minimum: 0.00000001

xcent The X-coordinate of the enclosing box.

Valid values: reals
Default value: **1e-07**

ycent The Y-coordinate of the enclosing box.

Valid values: reals
Default value: **1e-07**

zcent The Z-coordinate of the enclosing box.

Valid values: reals
Default value: **1e-07**

grow

Grows the currently selected fragment onto the currently selected grow bond.

Syntax:

grow

growbond

Specifies a grow bond for growing. A subsequent grow command will grow the currently selected fragment onto the this grow bond.

Syntax:

growbond ⟨ from_atom ⟩ ⟨ to_atom ⟩

Operands:

⟨ from_atom ⟩ ⟨ to_atom ⟩

The two atoms which define the grow bond. In the growing procedure the "to" atom will be replaced by a suitable atom from the incoming fragment.

growbond2

Specifies a second grow bond for growing. A subsequent grow command will grow the currently selected fragment onto the this grow bond.

Syntax:

growbond2 ⟨ from_atom ⟩ ⟨ to_atom ⟩

Operands:

⟨ from_atom ⟩ ⟨ to_atom ⟩

The two atoms which define the grow bond. In the growing procedure the "to" atom will be replaced by a suitable atom from the incoming fragment.

growdirection

Sets the grow direction in the current fragment mode. The direction name must be one of the valid directions for the current fragment mode.

Syntax:

growdirection ⟨direction_name⟩

Operands:

⟨direction_name⟩

The name of the grow direction which is to be made current. This must be a valid grow direction name within the current fragment.

growname

Set the growname to that specified for all atoms which match the ASL specification.

Syntax:

growname ⟨grow_name⟩ ⟨ASL⟩

Operands:

⟨grow_name⟩ ⟨ASL⟩

The first operand is the grow name which is to be applied to the atom. Only the first four characters of the growname will be used. The second operand is the ASL specification for all the atoms which are to have the growname applied.

happly

This is a standard alias for **hydrogenapply** (see [hydrogenapply], page 164).

hbondcriteria

Specify the criteria for calculating H-bonds in the workspace. The acceptor atom (A), H-bonded to the donor Hydrogen (H), is Oxygen, Nitrogen, Sulfur, or Fluorine.

Syntax:

hbondcriteria *acceptorangle*=⟨ x ⟩ *distance*=⟨ x ⟩ *donorangle*=⟨ x ⟩

Options:

acceptorangle
Minimum H — A-R angle, in degrees.

Valid values:	reals
Default value:	**90**
Minimum:	0.0
Maximum:	180.0

distance
Maximum separation between donor hydrogen atom and acceptor atom (H — A distance, in Angstroms).

Valid values:	reals
Default value:	**2.5**
Minimum:	0.0

donorangle
Minimum D-H — A angle, in degrees.

Valid values:	reals
Default value:	**120**
Minimum:	0.0
Maximum:	180.0

hbondset1

Specify the first set of atoms used in finding H-bonds in the 3D workspace.

Syntax:

hbondset1 ⟨ ASL ⟩

Operands:
⟨ ASL ⟩

A string in the atom specification language. Typical usage is to define hbondset1 and hbondset2. This set, hbondset1, defines the "from" atoms. The hbondset2 atoms define the "to" atoms. H-bonds are calculated between these two sets. That is, the H-bonds are inter-set H-bonds. No intra-set H-bonds are calculated. If hbondset2's ASL string is empty, then H-bonds are calculated for all atoms in hbondset1.

hbondset2

Specify the second set of atoms used in finding H-bonds in the 3D workspace.

Syntax:

hbondset2 ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

A string in the atom specification language. Typical usage is to define hbondset1 and hbondset2. This set, hbondset2, defines the "to" atoms. The hbondset1 atoms define the "from" atoms. H-bonds are calculated between these two sets. That is, the H-bonds are inter-set H-bonds. No intra-set H-bonds are calculated. If hbondset2's ASL string is empty, then H-bonds are calculated for all atoms in hbondset1.

help

This is a standard alias for **helpsearch** (see [helpsearch], page 154).

helpauto

Turns on or off the automatic popup help.

Syntax:

helpauto *delay*=⟨ n ⟩ on|off

Options:

delay The delay (in seconds) before the automatic help is popped up after the mouse enters the widget.

Valid values: integers
Default value: **2**

Operands:

on|off

If the operand is "on" then the popup help will be turned on. If off then it will be turned off.

helpcategory

Changes the current help category

Syntax:

helpcategory ⟨ category_name ⟩

Operands:

⟨ category_name ⟩

The name of the category for which the help topics are to be displayed.

helpsearch

Searches the help file for the specified search string. The search can be performed on all text or just the titles.

Syntax:

helpsearch *casesensitive*=yes | no *regularexpression*=yes | no *titleonly*=yes | no ⟨ search_string ⟩

Options:

casesensitive
: A boolean option which determines if the search is performed in a case sensitive manner
  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

regularexpression
: A boolean option which determines if the search string is to be treated as a regular expression
  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

titleonly
: A boolean option which determines if the search applies to the title of the help topics or the entire text
  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

Operands:
⟨search_string⟩
The text which the help file is to be searched for.
Aliases:
help (see [help], page 153)

helptopic

Loads the topic given by the operand

Syntax:

helptopic ⟨topic_name⟩

Operands:
⟨topic_name⟩
The name of the help topic to be displayed in the help panel. This must match the name of a topic exactly.

hideentries

Hide the selected entried in the project table and create a subset of unselected entries.This function also switches to the subset view.

Syntax:

hideentries

hidepanel

Hide the panel whose name is given by the operands.

Syntax:

hidepanel ⟨panel_name⟩

Operands:

⟨panel_name⟩

The name of the panel which is to be hidden. The names available for use in the "hidepanel" command are displayed in parentheses after each item in the main menu bar.

hidepanels

Hide all visible panels

Syntax:

hidepanels

hideproperty

This command hides the given property by creating a subset of all remaining properties in show state.This function also switch to the property subset view.

Syntax:

hideproperty ⟨propertyname⟩

Operands:

⟨propertyname⟩

The name of the property to hide.

historyvisible

Determines which types of commands will be visible in the command history list. Issuing a "historyvisible" alone will cause the list to be rebuilt with the current settings. A historyvisible all will make all commands in the list visible. A historyvisible readwrite=n will make all the read,write,sread and swrite commands hidden. Note that this only affects the visibility of the commands in the command history list of the script editor, not which commands are logged to a file.

Syntax:

historyvisible *1dplot*=yes | no *2dplot*=yes | no *adjust*=yes | no *alias*=yes | no *application*=yes | no *atomprop*=yes | no *beginundoblock*=yes | no *bondprop*=yes | no *build*=yes | no *clip*=yes | no *coloratom*=yes | no *debug*=yes | no *delete*=yes | no *displayatom*=yes | no *displayopt*=yes | no *entryexport*=yes | no *entryimport*=yes | no *entrywscreate*=yes | no *errorcheck*=yes | no *ffview*=yes | no *find*=yes | no *fit*=yes | no *glide*=yes | no *helpauto*=yes | no *helpcategory*=yes | no *helpsearch*=yes | no *helptopic*=yes | no *historyvisible*=yes | no *hold*=yes | no *htreat*=yes | no *impact*=yes | no *labelatom*=yes | no *liaison*=yes | no *ligrep*=yes | no *macromodel*=yes | no *measurements*=yes | no *monitor*=yes | no *pause*=yes | no *pausecommands*=yes | no *pose*=yes | no *prefer*=yes | no *print*=yes | no *project*=yes | no *qikprop*=yes | no *qsite*=yes | no *quit*=yes | no *rename*=yes | no *rep*=yes | no *ribbons*=yes | no *rotate*=yes | no *saveimage*=yes | no *savelayout*=yes | no *saverestoreview*=yes | no *script*=yes | no *sets*=yes | no *showhide*=yes | no *spotcenter*=yes | no *superimpose*=yes | no *system*=yes | no *table*=yes | no *tile*=yes | no *transformation*=yes | no *translate*=yes | no *undoredo*=yes | no *zoom*=yes | no [all]

Options:

1dplot

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

2dplot

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

adjust

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

alias

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

application

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

atomprop

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

beginundoblock
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

bondprop
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

build
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

clip
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

coloratom
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

debug
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

delete
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

displayatom
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

displayopt
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

entryexport
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

entryimport
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

entrywscreate
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

errorcheck
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

ffview

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

find

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

fit

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

glide

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

helpauto

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

helpcategory

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

helpsearch

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

helptopic

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

historyvisible

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

hold

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

htreat

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

impact

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

labelatom

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

liaison

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

ligrep

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

macromodel

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

measurements

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

monitor

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

pause

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

pausecommands

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

pose

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

prefer

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

print

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

project

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

qikprop

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

qsite

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

quit

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

rename

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

rep

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

ribbons

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

rotate

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

saveimage

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

savelayout

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

saverestoreview

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

script

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

sets

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

showhide

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

spotcenter

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

superimpose

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

system

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

table

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

tile

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

transformation

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

translate

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

undoredo

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

zoom

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

[all]

If "all" is specified then all commands will be visible in the command history list.

hold

The hold command permits holding the current structures off screen.

Syntax:

hold ⟨hold_name⟩

Operands:

⟨hold_name⟩

The name which will be applied to the hold set. This name will reference the hold in subsequent "addfromhold" and "replacefromhold" commands. If the name contains embedded spaces then it must be enclosed in quotation marks.

htreat

This is a standard alias for **hydrogentreat** (see [hydrogentreat], page 165).

hydrogenapply

Add or remove hydrogens and lone pairs to the atoms defined by the ASL operand according to the currently set treatment.

Syntax:

hydrogenapply ⟨ASL⟩

Operands:

⟨ASL⟩

A string in the atom specification language. All atoms which match this specification will have hydrogens added or removed to become consistent with the current treatment.

Aliases:

happly (see [happly], page 151)

hydrogentreat

Choose a treatment for hydrogen atom addition/deletion.

Syntax:

hydrogentreat ⟨treatment_name⟩

Operands:

⟨treatment_name⟩

The operand is the name of a treatment which is to be made current. Note: treatment names are defined in $SCHRODINGER/mmshare-vX.X/data/mmhtreat.ini. There are both long and short names for each treatments and either can be used.

Aliases:

htreat (see [htreat], page 164)

impactbufferedatom

Specifies an atom to be buffered in an Impact calculation.

Syntax:

impactbufferedatom ⟨atom_number⟩

Operands:

⟨atom_number⟩

The number of an atom which is to be treated as buffered during an Impact calculation.

impactbufferedset

Specifies a set of atoms to be buffered in an Impact calculation.

Syntax:

impactbufferedset ⟨ASL⟩

Operands:
⟨ASL⟩
The operand must be a valid string in the atom specification language.

impactconstraints

Used to set all the options associated with constraints in impact

Syntax:

impactconstraints *bonds*=yes | no *buffer_force*=⟨x⟩
hmcbonds=yes | no *shake_tolerance*=⟨x⟩ *solvent*=yes | no

Options:

bonds If true, then all bonds will be constrained during the Impact calculation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

buffer_force
The buffered atom force constant

Valid values: reals
Default value: **25**

hmcbonds If true, then all bonds will be constrained during the HMC calculation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

shake_tolerance
The tolerance for the SHAKE (or RATTLE) algorithms.

Valid values: reals
Default value: **1e-07**
Minimum: 0.0

solvent [NOTE: This option is no longer used.] If true, then all solvent molecules will be held rigid during the Impact calculation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

impactcontinuumsolvent

Used to set all the options associated with the contiunuum solvent in Impact.

Syntax:

impactcontinuumsolvent *pbfcutoff*=⟨ x ⟩ *resolution*=low | medium | high *sgbcutoff*=⟨ x ⟩ *type*=sgb | pbf

Options:

pbfcutoff The type displacement threshold for the PBF calculation. shake_tolerance=d

Valid values:	reals
Default value:	**0.1**
Minimum:	0.0

resolution The resolution for the PBF solvation calculation.

Valid values:	low medium high
Default value:	**low**

sgbcutoff The type displacement threshold for the SGB calculation.

Valid values:	reals
Default value:	**0.1**
Minimum:	0.0

type The type of continuum solvent which is to be used in the calculation.

Valid values:	sgb pbf
Default value:	**sgb**

impactdynamics

Settings associated with molecular dynamics simulations in Impact.

Syntax:

impactdynamics *effectivedensity*=⟨x⟩ *ensemble*=nvt | nve | npt *inittempgauss*=⟨x⟩ *initvelo*=yes | no *isothercomp*=⟨x⟩ *numbermdsteps*=⟨n⟩ *targetpress*=⟨x⟩ *targettemp*=⟨x⟩ *tautemp*=⟨x⟩ *tauvol*=⟨x⟩ *timestep*=⟨x⟩ *volumescaling*=centerofmass | atom

Options:

effectivedensity
The effective density used during an NPT Impact dynamics simulation.

Valid values: reals
Default value: **1**
Minimum: 0.0000000000001

ensemble The target ensemble to be achieved during an Impact dynamics simulation.

Valid values: nvt
nve
npt
Default value: **nvt**

inittempgauss
The temperature used to initialize the velocities from a Gaussian distribution during an NPT Impact dynamics simulation.

Valid values: reals
Default value: **298.15**
Minimum: 0.0000000000001

initvelo An option which determines if the velocities are to be initialized from a Gaussian distribution during an Impact dynamics simulation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

isothercomp
The solvent isothermal compressibility to be used during an NPT Impact dynamics simulation.

Valid values: reals
Default value: **4.96e-05**
Minimum: 0.0000000000001

numbermdsteps
The total number of time steps to be performed during an Impact dynamics simulation.

Valid values: integers
Default value: **100**

Minimum: 1

targetpress

The target pressure to be used during an NPT Impact dynamics simulation.

Valid values: reals
Default value: **1**

targettemp

The target temperature to be used during a NVT or NPT Impact dynamics simulation.

Valid values: reals
Default value: **298.15**
Minimum: 0.0000000001

tautemp The temperature relaxation time to be used during a NVT or NPT Impact dynamics simulation.

Valid values: reals
Default value: **0.01**
Minimum: 0.0000000001

tauvol The volume relaxation time to be used during an NPT Impact dynamics simulation.

Valid values: reals
Default value: **0.01**
Minimum: 0.0000000001

timestep The time step to be used (in ps) during an Impact dynamics simulation.

Valid values: reals
Default value: **0.001**
Minimum: 0.0000001

volumescaling

The method of volume scaling to be used during an NPT Impact Dynamics simulation.

Valid values: centerofmass
atom
Default value: **centerofmass**

impactfastmultipole

Used to set all the options associated with the fast multipole method in Impact.

Syntax:

impactfastmultipole *level*=⟨ n ⟩ *maximum*=⟨ n ⟩
smoothing=yes | no

Options:

level The level for the fast multipole method.

Valid values:	integers
Default value:	**2**
Minimum:	1

maximum The maximum for the fast multipole method.

Valid values:	integers
Default value:	**7**
Minimum:	4
Maximum:	20

smoothing Whether or not to use smoothing with the fast multipole method.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

impactfrozenatom

Specifies a single atom to be frozen at its current position in an Impact calculation.

Syntax:

impactfrozenatom ⟨ atom_number ⟩

Operands:
⟨ atom_number ⟩
The number of an atom which is to be treated as frozen during an Impact calculation.

impactfrozenset

Specifies a set of atoms to be frozen at their current positions in an Impact calculation.

Syntax:

impactfrozenset ⟨ASL⟩

Operands:
⟨ASL⟩
A string in the atom specification language which describes the set of atoms which are to be treated as frozen in an Impact calculation.

impacthybridmc

Settings associated with hybrid Monte Carlo simulations in Impact.

Syntax:

impacthybridmc *ncycles*=⟨n⟩ *nmdmc*=⟨n⟩ *timestep*=⟨x⟩

Options:

ncycles The total number of HMC cycles to be performed during an Impact hybrid Monte Carlo simulation.

Valid values:	integers
Default value:	**100**
Minimum:	1

nmdmc The number of MD steps per HMC cycle to be performed during an Impact hybrid Monte Carlo simulation.

Valid values:	integers
Default value:	**4**
Minimum:	1

timestep The time step to be used (in ps) during an Impact hybrid Monte Carlo simulation

Valid values:	reals
Default value:	**0.001**
Minimum:	0.0000001

impactjob

This keyword is used to set various options associated with running Impact jobs.

Syntax:

impactjob *host*=⟨text⟩ *incorporate*=append | replace | ignore *job*=⟨text⟩ *login*=⟨text⟩ *structure_source*=selected_entries | workspace | file

Options:

host
: The name of the host for the Impact job.

Valid values: text strings
Default value:

incorporate
: How the results are to be incorporated into the project. This can be done with replacement of the existing entries, by appending as new entries to the project or by ignoring the final results.

Valid values: append
replace
ignore
Default value: **append**

job
: The name for the Impact job.

Valid values: text strings
Default value: **impacttmp**

login
: The login name under which a Impact will be run.

Valid values: text strings
Default value:

structure_source
: Whether to use the selected entries in the current project or what is in the workspace as input for the job.

Valid values: selected_entries
workspace
file
Default value: **workspace**

impactmdparams

Used to set parameters associated with molecular dynamics simulations in Impact.

Syntax:

impactmdparams *every*=⟨ n ⟩ *fastfreq*=⟨ n ⟩ *integrator*=verlet | rrespa *mdstatistics*=yes | no *mediumfreq*=⟨ n ⟩ *nprint*=⟨ n ⟩ *slowfreq*=⟨ n ⟩ *stoprot*=yes | no *traj*=yes | no *trajfile*=⟨ text ⟩ *trajvelocities*=yes | no

Options:

every An option which determines how many MD steps between saving a frame of the trajectory for an Impact dynamics simulation.

Valid values: integers
Default value: **5**

fastfreq RRESPA frequency for the fast forces.

Valid values: integers
Default value: **4**
Minimum: 1

integrator Which integration technique is to be used during the dynamics simulation

Valid values: verlet
rrespa
Default value: **verlet**

mdstatistics
An option which determines if statistics are to be gathered during an Impact dynamics simulation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

mediumfreq
RRESPA frequency for the medium forces.

Valid values: integers
Default value: **2**
Minimum: 1

nprint The number of steps at which output will be written during an Impact dynamics simulation.

Valid values: integers
Default value: **5**
Minimum: 0

slowfreq RRESPA frequency for the slow forces.

Valid values: integers
Default value: **1**
Minimum: 1

stoprot An option which determines if overall rotation and translation is stopped during the dynamics simulation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

traj An option which determines if the trajectory is to be saved during an Impact dynamics simulation.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

trajfile The name of the trajectory file.
Valid values: text strings
Default value: **trajectory.trj**

trajvelocities
An option which determines if the velocities are to be saved during an Impact dynamics simulation.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

impactminimization

Used to set up a minimization in Impact.

Syntax:

impactminimization *algorithm*=newton | conjugate | steepest *convergence*=eadng | energy | gradient *cutoff_1*=⟨x⟩ *cutoff_2*=⟨x⟩ *energy_change*=⟨x⟩ *gradient*=⟨x⟩ *initial_step_size*=⟨x⟩ *maximum_step_size*=⟨x⟩ *ncycles*=⟨n⟩ *nfull_1*=⟨n⟩ *nfull_2*=⟨n⟩ *qsite_energychange*=⟨x⟩ *qsite_gradient*=⟨x⟩ *qsiteminialgorithm*=newton | conjugate | steepest

Options:

algorithm Which algorithm is to be used for the minimization
Valid values: newton
conjugate
steepest
Default value: **newton**

convergence
How convergence is to be established during the minimization.
Valid values: eadng
energy
gradient

Default value: **eadng**

cutoff_1 Minimization parameter for Truncated Newton. Long range force cutoff.

Valid values: reals
Default value: **10**
Minimum: 0.0

cutoff_2 Minimization parameter for Truncated Newton. Long range force cutoff.

Valid values: reals
Default value: **10**
Minimum: 0.0

energy_change
The energy change convergence criterion.

Valid values: reals
Default value: **1e-07**
Minimum: 0.0

gradient The gradient convergence criterion.

Valid values: reals
Default value: **0.01**
Minimum: 0.0

initial_step_size
The initial steps size for the minimization

Valid values: reals
Default value: **0.05**
Minimum: 0.0

maximum_step_size
The maximum step size for the minimization.

Valid values: reals
Default value: **1**
Minimum: 0.0

ncycles The maximum number of minimization cycles to be performed.

Valid values: integers
Default value: **100**

nfull_1 Minimization parameter for Truncated Newton. Update long range forces every X steps.

Valid values: integers
Default value: **10**
Minimum: 0

nfull_2 Minimization parameter for Truncated Newton. Update long range forces every X steps.

Valid values: integers
Default value: **10**
Minimum: 0

qsite_energychange
The energy change convergence criterion for QSite.

Valid values: reals
Default value: **0.1**
Minimum: 0.0

qsite_gradient
The gradient convergence criterion for QSite.

Valid values: reals
Default value: **0.2**
Minimum: 0.0

qsiteminialgorithm
Which algorithm is to be used for the minimization for qsite jobs.

Valid values: newton
conjugate
steepest
Default value: **conjugate**

impactperiodicboundary

Used to set all the options associated with the periodic boundary conditions in Impact.

Syntax:

impactperiodicboundary *alpha*=⟨ x ⟩ *ewald*=yes | no
kvectormax=⟨ n ⟩ *xsize*=⟨ x ⟩ *ysize*=⟨ x ⟩ *zsize*=⟨ x ⟩

Options:

alpha The alpha factor for the Ewald long-range correction.

Valid values: reals
Default value: **0.25**
Minimum: 0.0

ewald Whether or not to use the Ewald long-range correction with the fast multipole method.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

kvectormax
The maximum length of the k-space vectors in the Ewald long-range correction.

Valid values: integers
Default value: **5**
Minimum: 1

xsize The size of the box in the X-dimension.

Valid values: reals
Default value: **18.62**
Minimum: 18.62

ysize The size of the box in the Y-dimension.

Valid values: reals
Default value: **18.62**
Minimum: 18.62

zsize The size of the box in the Z-dimension.

Valid values: reals
Default value: **18.62**
Minimum: 18.62

impactpotential

Set various options associated with the definition of the potential energy to be used in a Impact job.

Syntax:

impactpotential *contiuum_solvent*=yes | no *dielectric*=⟨x⟩ *electrostatics*=constant | distance_dependant *fast_multipole*=yes | no *field*=oplsaa | opls1999 | opls2001 | opls2003 *paramfile*=⟨text⟩ *periodic_boundary*=yes | no *qsitefield*=oplsaa | opls1999 | opls2001 | opls2003 *truncate*=yes | no

Options:

contiuum_solvent
Whether or not to use the continuum solvent

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

dielectric The dielectric constant to be used in the electrostatic part of the energy calculation.

Valid values: reals
Default value: **1**
Minimum: 0.9999999999

electrostatics
The electrostatic treatment to be used in the Impact calculation.

Valid values: constant
distance_dependant
Default value: **constant**

fast_multipole
Whether or not to use the fast multipole method for electrostatic interactions.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

field The force field to be used for the Impact calculation.

Valid values: oplsaa
opls1999
opls2001
opls2003
Default value: **opls2001**

paramfile The name of the parameter file.

Valid values: text strings
Default value: **paramstd.dat**

periodic_boundary
Whether or not to use the periodic boundary conditions

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

qsitefield The force field to be used for the Impact calculation.

Valid values: oplsaa
opls1999
opls2001
opls2003
Default value: **opls2001**

truncate Whether or not to truncate the non-bonded interactions.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

impactstart

Start an Impact input file with the current settings.

Syntax:

impactstart

impacttask

Determines which impact task is currently being set up.

Syntax:

impacttask soak|mini|dynamics|hmc|qsite|glide|liaison

Operands:
soak|mini|dynamics|hmc|qsite|glide|liaison
The type of impact task which is to be set up. The operand cannot be abreviated and must be given in full.

impacttruncation

Used to set all the options associated with the truncation of non-bonded interactions in Impact.

Syntax:

impacttruncation *distance*=⟨ x ⟩ *updatefrequency*=⟨ n ⟩

Options:

distance The truncation distance for residue-based cutoffs.
Valid values: reals

Default value: **12**
Minimum: 0.0

updatefrequency
The number of steps between update of the neighbor list.

Valid values: integers
Default value: **10**
Minimum: 1

impactwrite

Write an Impact input file with the current settings.

Syntax:

impactwrite

invert

Inverts the chirality around a chiral atom.

Syntax:

invert ⟨chiral_atom⟩ ⟨non_moving_atom1⟩ ⟨non_moving_atom2⟩

Operands:
⟨chiral_atom⟩ ⟨non_moving_atom1⟩ ⟨non_moving_atom2⟩
Three atom numbers. The first is the atom around which the chirality is to be inverted. The second and third are atoms which are attached to the chiral atom but are not to be moved in the inversion process.

invertset

Inverts all chiral centers in the specified set of atoms.

Syntax:

invertset ⟨ASL⟩

Operands:
⟨ASL⟩
An ASL specification of the atoms which are to have their chiral centers inverted. Because of the way this works, this set of atom should involve at least whole molecules.

jaguarassignatomnames

Sets the atom names of the matching atoms as per the jaguar standards i.e, element name + atom number. Incase of any duplicates, integer part of the name is incremented till it is unique in the Workspace

Syntax:

jaguarassignatomnames ⟨ASL⟩

Operands:
⟨ASL⟩
The ASL expression describing the set of atoms which are to have their atom names made unique.

jaguarimportgeometry

Import structures into the current project from a Jaguar input file.

Syntax:

jaguarimportgeometry *all*=yes | no *end*=yes | no *start*=⟨n⟩ *total*=⟨n⟩ *wsinclude*=none | first | all *wsreplace*=yes | no [⟨filename⟩]

Options:

all This determines whether all structures will be imported, or just a specified range.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

end This determines if all structures in the file are to be imported starting from the structure specified by start.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

start This option sets the number of the first structure to be imported, if not importing all.

Valid values: integers
Default value: **1**
Minimum: 1

total The total number of structures to be imported from the file, if not importing all structures.

Valid values: integers
Default value: **1**
Minimum: 1

wsinclude This option determines which of the imported structures are to be included in the workspace. Valid values are "none", "first", or "all".

Valid values: none
first
all
Default value: **first**

wsreplace This determines whether the structures currently in the workspace will be replaced by the included imported structures, or whether they will be be included in the workspace also.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:

[⟨filename⟩]

The name of the file from which structures will be imported. If no name is specified, then no import will be done.

jaguarinputfilesjob

This keyword is used to set various options associated with running Jaguar input files jobs.

Syntax:

jaguarinputfilesjob *input_files*=⟨ text ⟩

Options:

input_files Specifies existing structure file to be used as input files for running the job

Valid values: text strings
Default value:

jaguarjob

This keyword is used to set various options associated with running Jaguar jobs.

Syntax:

jaguarjob *structure_files*=⟨ text ⟩ *structure_source*=selected_entries | workspace | file

Options:

structure_files
If structure_source is Selected structure files , specifies existing structure file to be used as input files for running the job.

Valid values: text strings
Default value:

structure_source
Whether to use the selected entries in the current project or what is in the workspace as input for the job.

Valid values: selected_entries
workspace
file
Default value: **workspace**

jaguarselectextendtablerow

Extends the selection to this row in the given orbital table in the Surfaces Property for Jaguar.

Syntax:

jaguarselectextendtablerow *table*=alpha | beta ⟨row⟩

Options:

table Indicates whether to select a row in the alpha or beta orbital table.

Valid values: alpha
beta
Default value: **alpha**

Operands:
⟨row⟩
The row number to extend the select to.

jaguarselectonlytablerow

Selects only this row in the given orbital table in the Surface Property for Jaguar.

Syntax:

jaguarselectonlytablerow *table*=alpha | beta ⟨row⟩

Options:

table Indicates whether to select a row in the alpha or beta orbital table.

Valid values: alpha
beta
Default value: **alpha**

Operands:
⟨row⟩
The row number to select only in the table row.

jobcleanup

Specifies a job to cleanup.

Syntax:

jobcleanup *files*=jobandmonitor | all ⟨job_id⟩

Options:

files This option determines which files are removed when the job is cleaned up. Either just the job record and monitoring files or all files associated with the job.

Valid values: jobandmonitor
all
Default value: **jobandmonitor**

Operands:

⟨job_id⟩
The ID of the job which is to be cleaned up.

jobsettings

This keyword is used to set various options associated with running a backend job.

Syntax:

jobsettings *directory*=⟨text⟩ *distributesubjobs*=maxprocessors | specifiedprocessors *host*=⟨text⟩ *incorporate*=appendentries | replaceentries | ignoreentries | ingoreentries *jobname*=⟨text⟩ *login*=⟨text⟩ *maxstructures*=⟨n⟩ *numcpus*=⟨n⟩ *numsubjobs*=⟨n⟩ *subjobsprocessors*=⟨n⟩ *tmpdir*=⟨text⟩ ⟨model_name⟩

Options:

directory The directory to write the output to.

Valid values: text strings
Default value:

distributesubjobs
How the subjobs are to be distributed over the available processors

Valid values: maxprocessors
specifiedprocessors
Default value: **maxprocessors**

host The name of the host for the backend job.

Valid values: text strings
Default value: **host**

incorporate
How the results are to be incorporated into the project. This can be done by appending as new entries to the project or by ignoring the final results.

Valid values: appendentries
replaceentries
ignoreentries
ingoreentries
Default value: **appendentries**

jobname The name for the backend job.

Valid values: text strings
Default value: **job**

login The login name under which the job will be run.

Valid values: text strings
Default value: **user**

maxstructures
The maximum number of structures to be incorporated into the project.

Valid values: integers
Default value: **100**
Minimum: 0

numcpus The number of CPUs to be used for jobs that support multiple CPUs.

Valid values: integers
Default value: **1**
Minimum: 1

numsubjobs
The number of subjobs this current job is to be run as.

Valid values: integers
Default value: **1**
Minimum: 1

subjobsprocessors
The number of processors to distribute the subjobs over

Valid values: integers
Default value: **1**
Minimum: 1

tmpdir The path name of the scratch directory.
Valid values: text strings
Default value:

Operands:
⟨ model_name ⟩
The name of a model which has job settings like host, login, and so on.

kill

This is a standard alias for **energykill** (see [energykill], page 87).

labelatom

Displays labels for the atoms specified by the ASL operand.

Syntax:

labelatom *1charge*=yes | no *2charge*=yes | no *acolor*=yes | no *anum*=yes | no *atomicnumber*=yes | no *atomname*=yes | no *atype*=yes | no *chain*=yes | no *chirality*=yes | no *cindex*=⟨ n ⟩ *color*=⟨ text ⟩ *element*=yes | no *entryname*=yes | no *font*=⟨ text ⟩ *formalcharge*=yes | no *growname*=yes | no *headings*=yes | no *inscode*=yes | no *molnum*=yes | no *molnumentry*=yes | no *numentry*=yes | no *nummol*=yes | no *pdbname*=yes | no *resname*=yes | no *resnum*=yes | no *separator*=⟨ text ⟩ *showlabel*=yes | no *stereochemistry*=yes | no *title*=yes | no *user*=yes | no *utext*=⟨ text ⟩ *xoffset*=⟨ x ⟩ *yoffset*=⟨ x ⟩ ⟨ ASL ⟩

Options:

1charge A boolean option which determines if the charge 1 value will be included in the atom label
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

2charge A boolean option which determines if the charge 2 value will be included in the atom label
Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

acolor A boolean option which determines if labels will be colored the same as the atoms they label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

anum A boolean option which determines if the atom numbers will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

atomicnumber
A boolean option which determines if the atomic number will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

atomname A boolean option which determines if the atom name will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

atype A boolean option which determines if the MacroModel atom types will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

chain A boolean option which determines if the chain name will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

chirality A boolean option which determines if the atom chirality (R or S) will be included in the label.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

cindex An integer which indicates color index which is to be used for the labels. This will be ignored unless the acolor option is off

Valid values: integers
Default value: **2**
Minimum: 1
Maximum: 128

color A string which is the color name for atom labels. This will be ignored unless the acolor option is off

Valid values: text strings
Default value:

element A boolean option which determines if the element symbol will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

entryname
A boolean option which determines if the entry name will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

font A string which determines which font is to be used for the labels

Valid values: text strings
Default value: **Arial Bold Small**

formalcharge
A boolean option which determines if the formal charge will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

growname A boolean option which determines if the grow name will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

headings A boolean option which determines if the label fields will have headers in the labels (anum=1, atype=C2 and so on).

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

inscode A boolean option which determines if the pdb residue insertion code will be included in the label string

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

molnum A boolean option which determines if the molecule number will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

molnumentry
A boolean option which determines if the molecule number by entry will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

numentry A boolean option which determines if the atom number by entry will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

nummol A boolean option which determines if the atom number by molecule will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

pdbname A boolean option which determines if the pdb atom name will be included in the label string

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

resname A boolean option which determines if the pdb residue name will be included in the label string

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

resnum A boolean option which determines if the residue number will be included in the label string

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

separator A string option which is the inserted between each field in the label string.

Valid values: text strings
Default value:

showlabel A boolean option which determines if an atom label is displayed when the atom itself is undisplayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

stereochemistry
A boolean option which determines if the double bond strerochemistry (E or Z) will be included in the label.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

title A boolean option which determines if the entry title will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

user A boolean option which determines if the user-defined text will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

utext This option sets the user defined text for atom labels

Valid values: text strings
Default value:

xoffset A double option which sets the x-offset from the atom center at which the labels are to appear.

Valid values: reals
Default value: **0.05**

yoffset A double option which sets the y-offset from the atom center at which the labels are to appear.

Valid values: reals
Default value: **-0.15**

Operands:
⟨ASL⟩
A string in the atom specification language. All atoms which match this specification will be labeled with the current label settings.

labelclear

Clears labels from the atoms specified by the ASL operand.

Syntax:

labelclear ⟨ASL⟩

Operands:
⟨ASL⟩
A string in the atom specification language. All atoms which match this specification will have their labels removed.

labelformat

Sets the label format string which will be used in subsequent labeling operations.

Syntax:

labelformat ⟨format_string⟩

Operands:

⟨format_string⟩

A string which defines the current labeling format. See on-line help for details.

labelupdate

Apply the current label format to all the atoms which are labeled.

Syntax:

labelupdate [NON_EL]

Operands:

[NON_EL]

A string which defines the relabeling method according to its caller (NON_EL for relabeling only atom labels, excluding the existing element labels. Otherwise both elementlabels and atom labels are relabelled).

liaisonanalysis

Used to do the analysis

Syntax:

liaisonanalysis *liaisonalpha*=⟨ x ⟩ *liaisonanalysistype*=fit | predict
liaisonbefile=⟨ text ⟩ *liaisonbeta*=⟨ x ⟩
liaisonenerymodeltype=equation | glidescore
liaisonfitname=⟨ text ⟩ *liaisonfixalpha*=yes | no
liaisonfixbeta=yes | no *liaisonfixgamma*=yes | no
liaisonfixintercept=yes | no *liaisonfixslope*=yes | no
liaisongamma=⟨ x ⟩ *liaisonintercept*=⟨ x ⟩ *liaisonlistfile*=⟨ text ⟩
liaisonslope=⟨ x ⟩ *liaisonspecligs*=readligs | enterligs

Options:

liaisonalpha
This option sets the van der Waals (Alpha) coefficient.
Valid values: reals
Default value: **0**

liaisonanalysistype
This option determines the analysis type, including two options of Predict (predict) and Fit (fit).
Valid values: fit
predict
Default value: **fit**

liaisonbefile
This option determines the name of the ligand binding energy file
Valid values: text strings
Default value:

liaisonbeta
This option sets the electrostatic (Beta) coefficient.
Valid values: reals
Default value: **0**

liaisonenerymodeltype
This option determines which binding enery model is used, LIA equation or Glidescore
Valid values: equation
glidescore
Default value: **equation**

liaisonfitname
This option determines the names of ligands to be predicted
Valid values: text strings
Default value:

liaisonfixalpha
Allows the alpha value to be used as a constraint.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

liaisonfixbeta
Allows the beta value to be used as a constraint.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

liaisonfixgamma
Allows the gamma value to be used as a constraint

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

liaisonfixintercept
Allows the Glidescore intercept to be used as a constraint

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

liaisonfixslope
Allows the glidescore slope to be used as a constraint.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

liaisongamma
This option sets cavity (Gamma) coefficient.

Valid values: reals
Default value: **0**

liaisonintercept
This option sets the Glidescore intercept

Valid values: reals
Default value: **0**

liaisonlistfile
This option determines the name of file that contains names of ligands to be predicted

Valid values: text strings
Default value:

liaisonslope
This option sets the Glidescore slope

Valid values: reals
Default value: **0**

liaisonspecligs
This option determines the methods used to specify the ligands either by reading ligand names from a text file (readligs), or entering a comma separated list of ligand names (enterligs).

Valid values: readligs
enterligs
Default value: **readligs**

liaisonparameters

Used to specify simulation parameters in a Liasion (Linear Interaction Approximation).

Syntax:

liaisonparameters *algorithm*=newton | conjugate | steepest
liaisonboundgcut=⟨ x ⟩ *liaisonboundheat*=⟨ x ⟩
liaisonboundint=⟨ n ⟩ *liaisonboundmini*=⟨ n ⟩
liaisonboundpremini=⟨ n ⟩ *liaisonboundprod*=⟨ x ⟩
liaisonfreeint=⟨ n ⟩ *liaisonliggcut*=⟨ x ⟩ *liaisonligheat*=⟨ x ⟩
liaisonligmini=⟨ n ⟩ *liaisonligpremini*=⟨ n ⟩ *liaisonligprod*=⟨ x ⟩
liaisonmethod=mini | hmc | dyn *liaisonreltime*=⟨ x ⟩
liaisonrescut=⟨ x ⟩ *liaisontemp*=⟨ x ⟩ *minibound*=yes | no
minilig=yes | no

Options:

algorithm Which algorithm is to be used for the minimization
Valid values: newton
conjugate
steepest
Default value: **newton**

liaisonboundgcut
This option determines the RMS gradient for convergence (in kcal/mol/A)
Valid values: reals
Default value: **0.05**

liaisonboundheat
This option determines the time for heating/ equilibration (ps)
Valid values: reals
Default value: **5**
Minimum: 0.0

liaisonboundint
This option determines the number of steps for collecting Liaison statistics

Valid values: integers
Default value: **10**
Minimum: 0

liaisonboundmini
This option determines the maximum number of minimization steps

Valid values: integers
Default value: **500**
Minimum: 0

liaisonboundpremini
This option determines the maximum number of minimization steps

Valid values: integers
Default value: **500**
Minimum: 0

liaisonboundprod
This option determines the time for data collection (in ps)

Valid values: reals
Default value: **5**
Minimum: 0.0

liaisonfreeint
This option determines the number of steps for collecting Liasion statistics

Valid values: integers
Default value: **10**
Minimum: 0

liaisonliggcut
This option determines the RMS gradient for convergence

Valid values: reals
Default value: **0.01**
Minimum: 0.0

liaisonligheat
This option determines the time for heating/ equilibration (ps)

Valid values: reals
Default value: **5**
Minimum: 0.0

liaisonligmini
This option determines the maximum number of minimization steps

Valid values: integers

Default value: **500**
Minimum: 0

liaisonligpremini

This option determines maximum number of minimization steps

Valid values: integers
Default value: **1000**
Minimum: 0

liaisonligprod

This option determines the time for data collection (in ps)

Valid values: reals
Default value: **5**
Minimum: 0.0

liaisonmethod

This option determines which sampling method will be used during a Liaison analysis. The 3 options are minimization (mini), Hybrid Monte Carlo (hmc) and Molecular dynamics (dyn).

Valid values: mini
hmc
dyn
Default value: **mini**

liaisonreltime

This option determines the temperature relaxation time

Valid values: reals
Default value: **0.01**
Minimum: 0.0

liaisonrescut

This option determines the residue-based cutoff for nonbonded interactions

Valid values: reals
Default value: **15**
Minimum: 0.0

liaisontemp

This option determines the simulation temperature

Valid values: reals
Default value: **300**
Minimum: 0.0

minibound

This option determines whether the ligand- receptor complex is minimized before running simulation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

minilig This option determines whether the ligand is minimized before running the simulation

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

liaisonselectlig

Defines a on-screen molecule to be treated as the ligand for a Liaison calculation

Syntax:

liaisonselectlig ⟨ molecule_number ⟩

Operands:
⟨ molecule_number ⟩
The number of a molecule to be included as the ligand.

liaisonsettings

Used to set values associated with the Linear Interaction Approximation (Liaison)

Syntax:

liaisonsettings *liaisonsubdir*=⟨ text ⟩ *liaisontype*=sim | analyze *numproc*=⟨ n ⟩

Options:

liaisonsubdir
This option determines the subdirectory to use for Liaison jobs.

Valid values: text strings
Default value: **liaison**

liaisontype
This option determines the job type for Liaison analysis. There are two types: (1) Simulate (sim); (2) Analyze results of earlier simulations (analyze)

Valid values: sim
analyze
Default value: **sim**

numproc This option determines the number of processors the job is to be run on.

Valid values: integers
Default value: **1**
Minimum: 1

liaisonsystem

Used to specify receptor and ligand(s) to be simulated in the Linear Interaction Approximation (Liaison)

Syntax:

liaisonsystem *haslialig*=yes | no *liaisonligfmt*=m2io | sd | mol2 | pdb *liaisonligfname*=⟨ text ⟩ *liaisonsimtype*=mult | single *liaisonsname*=⟨ text ⟩ *ligftype*=ligstruct | liglist *usepartialcharges*=yes | no

Options:

haslialig This option determines whether the displayed structure includes the ligands

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

liaisonligfmt
This option for selecting the format of the ligand file. The four possible formats are MAESTRO (m2io), MDL SD (sd), MOL2 (mol2) and PDB (pdb)

Valid values: m2io
sd
mol2
pdb
Default value: **m2io**

liaisonligfname
This option determines the name of the ligand file

Valid values: text strings
Default value:

liaisonsimtype
This option determines the type of ligand(s)-receptor structures to be simulated, having two options: (1) Multiple ligands, single receptor; (2) Single ligand, single receptor.

Valid values: mult
single
Default value: **mult**

liaisonsname
This option determines the name to be used for this ligand

Valid values: text strings
Default value: **Lig**

ligftype This option detemines the source of ligand structure(s), having two options: (1) File containing a single ligand structure (ligstruct); and (2) File containing a list of ligand names and structure files (liglist).

Valid values: ligstruct
liglist
Default value: **ligstruct**

usepartialcharges
This option determines whether the to use partial charges from the Maesro file

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ligandbond

Specifies two atoms which define a bond to be treated as a ligand bond during a conformational search.

Syntax:

ligandbond ⟨atom1⟩ ⟨atom2⟩

Operands:
⟨atom1⟩ ⟨atom2⟩
The numbers of two atoms which are to be treated as a ligand bond during a conformational search. Note that specifying a-b is the same as specifying b-a.

lightambient

Sets the ambient intensity of the light.

Syntax:

lightambient *intensity*=⟨ x ⟩

Options:

intensity Ambient lighting component.

Valid values:	reals
Default value:	**0.1**
Minimum:	0.0
Maximum:	1.0

lightdiffuse

Sets the diffuse intensity of the light.

Syntax:

lightdiffuse *intensity*=⟨ x ⟩

Options:

intensity Diffuse lighting component.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

lightposition

Sets the position of the light.

Syntax:

lightposition ⟨x⟩ ⟨y⟩ ⟨z⟩

Operands:
⟨x⟩ ⟨y⟩ ⟨z⟩
The position of the light.

lightspecular

Sets the specular intensity of the light.

Syntax:

lightspecular *intensity*=⟨x⟩

Options:

intensity Specular lighting component.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

ligprep

This keyword is used to set various options associated with running LigPrep jobs.

Syntax:

ligprep *desalt*=yes | no *gen_conform*=⟨n⟩ *gen_stereo*=⟨n⟩ *gen_tautomers*=yes | no *input_file*=⟨text⟩ *ionization*=generate | neutralize | retain *output_format*=maestro | sdf *ph*=⟨x⟩ *ph_tolerance*=⟨x⟩ *stereoisomers*=retain | determine | generate *structure_source*=selected_entries | workspace | file

Options:

desalt Desalt

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

gen_conform
Generate low energy ring confirmations

Valid values: integers
Default value: **1**
Minimum: 1

gen_stereo Generate stereoisomers (maximum)

Valid values: integers
Default value: **32**
Minimum: 1

gen_tautomers
Generate tautomers

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

input_file The name of the structure input file.

Valid values: text strings
Default value:

ionization How to generate the ionization states

Valid values: generate
neutralize
retain
Default value: **generate**

output_format
Whether to output the file in Maestro or SDF format.

Valid values: maestro
sdf
Default value: **maestro**

ph Ionization pH

Valid values: reals
Default value: **7**
Minimum: 0.0
Maximum: 14.0

ph_tolerance
Ionization pH tolerance

Valid values: reals
Default value: **2**
Minimum: 0.0
Maximum: 7.0

stereoisomers
How to generate the stereoisomers

Valid values: retain
determine
generate
Default value: **retain**

structure_source
Whether to use the selected entries in the current project, or what is in the workspace, or a specified file with multiple structures as structure input for the job.

Valid values: selected_entries
workspace
file
Default value: **file**

ligprepstart

Start a LigPrep job with the current settings.

Syntax:

ligprepstart

localatoms

Specify the group of local (transformation) atoms

Syntax:

localatoms ⟨ASL⟩

Operands:
⟨ASL⟩
For specifying which atoms are to be locally transformed. A list of atoms is created at the time the command is issued. Use the transform command to specify local scope, then use the rotate and/or translate commands to perform local transformations. To get back to global scope use the transform command to set the scope back to global.

localbitset

Specify whether local bitsets (atoms and center) are updated whenever contents of Workspace change. Unless the lock is enabled, these bitsets are re-evaluated each time atoms are changed, including changes in atom coordinates due to local adjustment.

Syntax:

localbitset *lock*=yes | no

Options:

lock Enable/disable lock on local bitsets for as long as ASL expressions and number of atoms in Workspace remain unchanged.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

localcenter

Set local transformation center to given centroid

Syntax:

localcenter

logfile

This is a standard alias for **scriptlogfile** (see [scriptlogfile], page 383).

logp

Support calculation of LogP of MacroModel.

Syntax:

logp *enable*=yes | no *secsolvent*=none | water | chcl3 | octanol

Options:

enable Determines whether logp will be used with this multiple minimization.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

secsolvent This option determines the second solvent for logp. softlimit=<n> <input_file_name>

Valid values: none
water
chcl3
octanol
Default value: **water**

macrodefine

Define a macro. Definition is a semicolon separated string of maestro commands.

Syntax:

macrodefine ⟨macro_name⟩ ⟨definition⟩

Operands:
⟨macro_name⟩ ⟨definition⟩
The first operand is the name of the macro. If this contains embedded spaces then it must be enclosed in double quotes. The second operand is the definition. If this contains embedded spaces then it must be enclosed in double quotes. To run the macro, execute macrorun <macro_name>

macrorun

Invokes an already defined macro

Syntax:

macrorun ⟨ macro_name ⟩

Operands:

⟨ macro_name ⟩

The operand is the name of the macro. If this contains embedded spaces then it must be enclosed in double quotes. As a shortcut macros can also be run without preceding them with macrorun . Using macrorun has the advantage that if the macro doesn't exist the program will tell you the macro doesn't exist. If you omit macrorun and Maestro cannot find the macro, then Maestro will report that the command does not exist. This will be misleading and possibly confusing because it's really the macro that will not exist.

materialmolecular

Syntax:

materialmolecular *shininess*=⟨ x ⟩ *specular*=⟨ x ⟩

Options:

shininess Shininess component for material of molecules

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0
Maximum:	1.0

specular Specular component for material of molecules

Valid values:	reals
Default value:	**0.7**
Minimum:	0.0
Maximum:	1.0

mcsd

A command which defines settings for the MC/SD (Monte Carlo/Stochastic Dynamics) simulation.

Syntax:

mcsd *maxdof*=⟨ n ⟩ *mindof*=⟨ n ⟩ *ratio*=⟨ n ⟩ *temperature*=⟨ x ⟩

Options:

maxdof The maximum number of torsions which will will be varied at each MC trial.

Valid values: integers
Default value: **1**
Minimum: 1

mindof The minimum number of torsions which will will be varied at each MC trial.

Valid values: integers
Default value: **1**
Minimum: 1

ratio The ratio of Monte Carlo to Stochastic Dynamics steps in the simulation.

Valid values: integers
Default value: **1**
Minimum: 1

temperature
The temperature at which the MCSD simulation is to be performed. [NOTE: This option is no longer used.]

Valid values: reals
Default value: **300**
Minimum: 0.0

mini

This is a standard alias for **minienergy** (see [minienergy], page 208).

minienergy

Used to set values associated with a MacroModel energy minimization

Syntax:

minienergy *converge*=nothing | energy | gradient | movement
maxiter=⟨ n ⟩ *method*=sd | prcg | osvm | fmnr | tncg | lbfgs
threshold=⟨ x ⟩

Options:

converge This option determines which convergence criterion will be used during an energy minimization.

Valid values:	nothing energy gradient movement
Default value:	**gradient**

maxiter This option determines the maximum number of iterations which will be performed during an energy minimization.

Valid values:	integers
Default value:	**500**
Minimum:	0
Maximum:	9999999

method This option determines which minimization method will be used.

Valid values:	sd prcg osvm fmnr tncg lbfgs
Default value:	**prcg**

threshold This option determines what the convergence threshold will be.

Valid values:	reals
Default value:	**0.05**
Minimum:	0.0

Aliases:

mini (see [mini], page 208)

minta

Used to set up options associated with BatchMin molecular energy calculation.

Syntax:

minta *hardlimit*=⟨ x ⟩ *numenergy*=⟨ n ⟩ *numint*=⟨ n ⟩ *softlimit*=⟨ n ⟩ *temperature*=⟨ x ⟩ ⟨ input_file_name ⟩

Options:

hardlimit The hard limit for sampling along normal modes.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	3.0

numenergy
This option determines the number of energy evaluations per MINTA integration.

Valid values:	integers
Default value:	**2000**
Minimum:	1

numint This option determines the number of MINTA iteration.

Valid values:	integers
Default value:	**5**
Minimum:	1

softlimit The soft limit for sampling along normal modes.

Valid values:	integers
Default value:	**3**
Minimum:	1
Maximum:	3

temperature
The temperature for MINTA calculation.

Valid values:	reals
Default value:	**300**
Minimum:	0.0

Operands:

⟨ input_file_name ⟩

The name of the input file. This name must be given in full, including any suffix.

monitor

This is a standard alias for **energymonitor** (see [energymonitor], page 87).

monitorangle

Specifies a triplet of atoms to have their bond angle monitored during a MacroModel dynamics simulation.

Syntax:

monitorangle ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩

The atom numbers of three atoms which are to have the angle between them monitored during a dynamics simulation. Note that specifying a-b-c is the same as c-b-a

monitordistance

Specifies a pair of atoms to have their distance monitored during a MacroModel dynamics simulation.

Syntax:

monitordistance ⟨ atom1 ⟩ ⟨ atom2 ⟩

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩

The atom numbers of two atoms which are to have the distance between them monitored during a dynamics simulation. Note that specifying a-b is the same as b-a

monitorhbond

Specifies a quartet of atoms which define an H-bond to be monitored during a dynamics simulation. The atoms are specified as X-H...Y-Z e.g. N2-H3...O2=C2 for a amide-amide bond.

Syntax:

monitorhbond *distance*=⟨ x ⟩ *hyzangle*=⟨ x ⟩ *xhyangle*=⟨ x ⟩
⟨ xatom ⟩ ⟨ hatom ⟩ ⟨ yatom ⟩ ⟨ zatom ⟩

Options:

distance Specifies the maximum H...Y distance for an acceptable hydrogen bond.

Valid values:	reals
Default value:	**2.5**
Minimum:	0.0

hyzangle Specifies the minimum H...Y-Z angle for an acceptable hydrogen bond.

Valid values:	reals
Default value:	**90**
Minimum:	0.0
Maximum:	180.0

xhyangle Specifies the minimum X-H...Y angle for an acceptable hydrogen bond.

Valid values:	reals
Default value:	**120**
Minimum:	0.0
Maximum:	180.0

Operands:

⟨ xatom ⟩ ⟨ hatom ⟩ ⟨ yatom ⟩ ⟨ zatom ⟩

The first operand is the atom number of the heavy atom (X) to which the donor hydrogen is attached. The second operand is the atom number of the donor hydrogen itself (H). The third operand is the atom number of acceptor atom (Y) and the fourth operand is the atom number of the heavy atom (Z) attached to the acceptor.

monitorsetsurf

Specifies a set of atoms to be have their surface areas monitored during a MacroModel molecular dynamics simulation.

Syntax:

monitorsetsurf ⟨ASL⟩

Operands:
⟨ASL⟩
A string in the atom specification language which describes the set of atom which are to have their surface areas monitored during a dynamics simulation.

monitorsettings

Controls settings for how the monitor panel in Maestro functions. At present it's just for whether jobs for the current project are displayed or all jobs.

Syntax:

monitorsettings *onlycurrentproj*=yes | no

Options:

onlycurrentproj
This determines whether the jobs listed in the monitor panel are limited to those for the current project or all known jobs for the current user.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

monitorsurf

Specifies a single atom to have its surface area monitored during a MacroModel molecular dynamics simulation.

Syntax:

monitorsurf ⟨atom_number⟩

Operands:
⟨atom_number⟩
The number of an atom which is to have its surface areas monitored during a dynamics simulation.

monitortorsion

Specifies a quartet of atoms to have their torsion angle monitored during a MacroModel dynamics simulation.

Syntax:

monitortorsion ⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩

Operands:
⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩
The atom numbers of four atoms which are to have the torsion between them monitored during a dynamics simulation. Note that specifying a-b-c-d is the same as d-c-b-a

mouse

Specify how mouse movements are to be interpreted

Syntax:

mouse *rotate*=xy | x | y | z | adjust *translate*=xy | x | y | z
There are no operands for this command.

Options:

rotate How mouse movements are interpreted for rotation. Default is rotate in xy.
Valid values: xy
x
y
z
adjust

Default value: **xy**

translate How mouse movements are interpreted for translation. Default is translate in xy.

Valid values: xy
x
y
z

Default value: **xy**

Operands:
There are no operands for this command.

move

Moves the specified atoms by the offsets specified in x, y and z values

Syntax:

move ⟨atom_num⟩ ⟨xinc⟩ ⟨yinc⟩ ⟨zinc⟩

Operands:
⟨atom_num⟩ ⟨xinc⟩ ⟨yinc⟩ ⟨zinc⟩
The first operand is the number of the atom which is to be moved. The following three real numbers represent the offsets to be applied to the x, y and z coordinates in Angstroms.

multiplemini

Used to set up a multiple minimization. The operand is the name of a file which contains a number of structures.

Syntax:

multiplemini *maxdist*=⟨x⟩ *window*=⟨x⟩ ⟨input_file_name⟩

Options:

maxdist Maximum distance between atoms in equal structures.

Valid values: reals

Default value: **0.25**
Minimum: 0.0

window Specifies the energy window (in kJ/mol) for saving structures during a multiple minimization.

Valid values: reals
Default value: **50**
Minimum: 0.0

Operands:

⟨ input_file_name ⟩

The name of the input file. This name will contain all the structures which are to be minimized. This name must be given in full, including any suffix.

mutate

Replace the sidechain of any residue in set <ASL> with the sidechain of the currently selected fragment

Syntax:

mutate ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

The set of atoms which are to be mutated. Each residue represented in this set will be mutated.

nextpose

If there is more than one pose structure in the pose viewer, display the one after the last displayed pose. Upon reaching the end of the list, wraps around to the beginning.

Syntax:

nextpose

optimizefog

Fit clipping planes to displayed structure to maximize the shading contrast between near and far atoms.

Syntax:

optimizefog

partialcharge

Set the partial atomic charges for all atoms which match the ASL specification.

Syntax:

partialcharge ⟨ charge1 ⟩ ⟨ charge2 ⟩ ⟨ ASL ⟩

Operands:

⟨ charge1 ⟩ ⟨ charge2 ⟩ ⟨ ASL ⟩

The first operand must be a valid real number which will be used for the charge1 field of all matching atoms. The second operand must be a valid real number which will be used for the charge2 field of all matching atoms. The final operand is a valid ASL string which specifies which atoms are to have their charges changed.

pause

Pauses immediately for the length of time specified (in seconds, unless other units are specified). Setting a pause of zero (e.g. "0" or "0.00") will mean there is no pause. If no operand is specified, or a negative time is given, then Maestro will remain paused indefinitely, until it is interrupted. All pauses can be interrupted by pressing any key on the keyboard or any mouse button.

Syntax:

pause ⟨ pause_time ⟩ [sec | min | hour | day]

Operands:
⟨ pause_time ⟩ [sec | min | hour | day]
The length of time to pause - floating point value in seconds, accurate to within about 0.05 second.

pausecommands

Sets the time to pause after execution of every command in script (in seconds, unless other units are specified). Setting a pause of zero (e.g. "0" or "0.00") will mean there is no pause after each command. If no operand is specified, or a negative time is given, then the pause after each command will be indefinite, pausing until interrupted. All pauses can be interrupted by pressing any key on the keyboard or any mouse button.

Syntax:

pausecommands ⟨ pause_time ⟩ [sec | min | hour | day]

Operands:
⟨ pause_time ⟩ [sec | min | hour | day]
The time to pause after each command - floating point value in seconds, accurate to within about 0.05 second.

phaseaddcustomfeature

Adds the given custom feature.

Syntax:

phaseaddcustomfeature *code*=⟨ text ⟩ ⟨ name ⟩

Options:

code The name of the custom feature to add.
Valid values: text strings
Default value: **A**

Operands:
⟨ name ⟩
The name for the feature.

phaseaddligands

Adds the given ligands to the table.

Syntax:

phaseaddligands *activity*=⟨ text ⟩ ⟨ ESL ⟩

Options:

activity This determines which property (if any) to use as the activity property for the ligands.
Valid values: text strings
Default value:

Operands:
⟨ ESL ⟩
The entries to add as ligands.

phaseaddligandsfromrun

Adds the given ligands to the table.

Syntax:

phaseaddligandsfromrun *runname*=⟨ text ⟩ ⟨ ligand_names ⟩

Options:

runname This determines which run to use as the reference to copy the corresponding properties of the ligands.
Valid values: text strings
Default value:

Operands:
⟨ ligand_names ⟩
The ligand names to be added separated by semi-colon.

phasebuildqsar

Sets the QSAR options and launchs a Build QSAR job.

Syntax:

phasebuildqsar *gridspacing*=⟨ x ⟩ *logactivities*=yes | no
maxplsfactor=⟨ n ⟩

Options:

gridspacing
The grid spacing to be for the QSAR model grid. The valid range is 0.5 to 2.0 angstrom.

Valid values:	reals
Default value:	**1**
Minimum:	0.5
Maximum:	2.0

logactivities
This options is used to log all the activities.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

maxplsfactor
This option value range depends on the ligands in the training set. The valid range is 1 to N/5, where N is the number of ligands in the training set. If N is less than 5, the maximum number of factors is 1, but no QSAR model can be constructed.

Valid values:	integers
Default value:	**1**
Minimum:	1

phasechooseactiveset

Sets the Active flag based on whether the activity value is larger than the cutoff value.

Syntax:

phasechooseactiveset ⟨cutoff⟩

Operands:
⟨cutoff⟩
The cutoff value for deciding between active and inactive.

phasecleanupstructures

Launches a Cleanup Structures job for Phase.

Syntax:

phasecleanupstructures

phaseclearcentroidatoms

Removes all atoms from the centroid list for excluded volumes.

Syntax:

phaseclearcentroidatoms

phaseconfgen

Defines settings for Phase Generate Conformers job.

Syntax:

phaseconfgen *field*=mmffs *incorporate*=append | replace | ignore *maxdist*=⟨ x ⟩ *numsteps*=⟨ n ⟩ *postmaxiter*=⟨ n ⟩ *postprocessing*=yes | no *premaxiter*=⟨ n ⟩ *sampling*=standard | rapid | complete | thorough *solvation*=gbsa | distance_dependent *window*=⟨ x ⟩

Options:

field
This determines which force field mmffs|mmff|opls2001 is used. Currently we always use mmffs, so it will have only one option value.

Valid values: mmffs
Default value: **mmffs**

incorporate
This option controls the incorporation of the results (replace or append).

Valid values: append
replace
ignore
Default value: **replace**

maxdist
Maximum distance between atoms in equal structures.

Valid values: reals
Default value: **2**
Minimum: 0.0

numsteps
An option which sets the number of steps which will be performed during the conformational search. This also limits number of conformations generated.

Valid values: integers
Default value: **1000**
Minimum: 0

postmaxiter
This option determines the maximum number of iterations for post-minimization of generated structures.

Valid values: integers
Default value: **50**
Minimum: 0
Maximum: 9999999

postprocessing
Indicates whether or not to perform MacroModel postprocessing.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

premaxiter
This option determines the maximum number of iterations for pre-minimization of input structures.

Valid values:	integers
Default value:	**100**
Minimum:	0
Maximum:	9999999

sampling This determines whether rapid (standard) or thorough (complete) sampling will be used.

Valid values:	standard rapid complete thorough
Default value:	**standard**

solvation This determines whether GB/SA Water (gbsa) or Distance Dependent Dielectric (distance_dependent) solvation treatment is used.

Valid values:	gbsa distance_dependent
Default value:	**distance_dependent**

window The energy window (in kcal/mol) within which structures will be saved.

Valid values:	reals
Default value:	**10**
Minimum:	0.0

phasecreateexcludedvolume

Creates an excluded volume from the current centroid atoms.

Syntax:

phasecreateexcludedvolume

phasedbcleanupstructures

Launches a Cleanup Structures job for a Phase database.

Syntax:

phasedbcleanupstructures *alreadycleaned*=yes | no
ionization=retain | neutralize | generate *maxisomers*=⟨ n ⟩
stereoisomers=retain | determine | all *target_ph*=⟨ x ⟩
⟨ job name ⟩

Options:

alreadycleaned
Indicates whether or not the structures have already been cleaned.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ionization Sets the handling of ionization.

Valid values: retain
neutralize
generate
Default value: **retain**

maxisomers
The maximum number of stereoisomers to return for any ligand.

Valid values: integers
Default value: **10**
Minimum: 1

stereoisomers
Sets the handling of chiralities.

Valid values: retain
determine
all
Default value: **all**

target_ph The target pH for generating ionization states.

Valid values: reals
Default value: **7**

Operands:

⟨ job name ⟩

The name of the Cleanup Structures job.

phasedbclose

Close the current Phase 3D database.

Syntax:

phasedbclose

phasedbconfgen

Defines settings for PhaseDB Generate Conformers job.

Syntax:

phasedbconfgen *field*=mmffs *incorporate*=append | replace | ignore *maxdist*=⟨ x ⟩ *numsteps*=⟨ n ⟩ *postmaxiter*=⟨ n ⟩ *postprocessing*=yes | no *premaxiter*=⟨ n ⟩ *sampling*=standard | rapid | complete | thorough *solvation*=gbsa | distance_dependent *window*=⟨ x ⟩

Options:

field This determines which force field mmffs|mmff|opls2001 is used. Currently we always use mmffs, so it will have only one option value.

Valid values: mmffs
Default value: **mmffs**

incorporate
This option controls the incorporation of the results (replace or append).

Valid values: append
replace
ignore
Default value: **replace**

maxdist Maximum distance between atoms in equal structures.

Valid values: reals
Default value: **2**
Minimum: 0.0

numsteps An option which sets the number of steps which will be performed during the conformational search. This also limits number of conformations generated.

Valid values: integers
Default value: **100**
Minimum: 0

postmaxiter
This option determines the maximum number of iterations for post-minimization of generated structures.

Valid values: integers
Default value: **50**
Minimum: 0
Maximum: 9999999

postprocessing
Indicates whether or not to perform MacroModel postprocessing.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

premaxiter
This option determines the maximum number of iterations for pre-minimization of input structures.

Valid values: integers
Default value: **100**
Minimum: 0
Maximum: 9999999

sampling This determines whether rapid (standard) or thorough (complete) sampling will be used.

Valid values: standard
rapid
complete
thorough
Default value: **standard**

solvation This determines whether GB/SA Water (gbsa) or Distance Dependent Dielectric (distance_dependent) solvation treatment is used.

Valid values: gbsa
distance_dependent
Default value: **distance_dependent**

window The energy window (in kcal/mol) within which structures will be saved.

Valid values: reals
Default value: **10**
Minimum: 0.0

phasedbcreatesites

Launches a Create Sites job for a Phase database.

Syntax:

phasedbcreatesites ⟨job name⟩

Operands:
⟨job name⟩
The name of the Create Sites job.

phasedbcreatesubset

Creates a subset from the given file.

Syntax:

phasedbcreatesubset *file*=⟨text⟩ ⟨subset name⟩

Options:

file The name of the text file which contains the ligand names.
Valid values: text strings
Default value:

Operands:
⟨subset name⟩
The name of the subset to create.

phasedbdelete

Delete the current Phase 3D database.

Syntax:

phasedbdelete

phasedbdeletesubset

Deletes the currently selected subset.

Syntax:

phasedbdeletesubset

phasedbgenerateconformers

Launches a Generate Conformers job for a Phase database.

Syntax:

phasedbgenerateconformers

phasedbimport

Imports an existing Phase 3D database into the current database.

Syntax:

phasedbimport ⟨db_dir_path⟩

Operands:
⟨db_dir_path⟩
The path (name and location) of the Phase 3D database directory to be imported.

phasedbimportfile

Imports a feature definitions file for use in a Create Database job.

Syntax:

phasedbimportfile ⟨file⟩

Operands:

⟨file⟩

The file to import the feature definitions from.

phasedbimportrun

Imports a feature definitions file from the given run for use in a Create Database job.

Syntax:

phasedbimportrun ⟨run⟩

Operands:

⟨run⟩

The run to import the feature definitions from.

phasedbnewfromfile

Create a new Phase 3D database and open it in Maestro.

Syntax:

phasedbnewfromfile ⟨db_dir_path⟩ ⟨struct_file_path⟩

Operands:

⟨db_dir_path⟩ ⟨struct_file_path⟩

The path (name and location) of the directory to be created as the new Phase 3D database directory and the path for the (maestro or sd) structure file used to initialize it.

phasedbopen

Open an existing Phase 3D database into Maestro.

Syntax:

phasedbopen ⟨db_dir_path⟩

Operands:
⟨db_dir_path⟩
The path (name and location) of the Phase 3D database directory to be opened.

phasedbrenamesubset

Renames the given subset row in the table.

Syntax:

phasedbrenamesubset *name*=⟨text⟩ ⟨row number⟩

Options:

name The new name for the subset.
Valid values: text strings
Default value:

Operands:
⟨row number⟩
The row number of the subset to rename.

phasedbresetall

Resets the feature definitions for use in a Create Database job.

Syntax:

phasedbresetall

phasedbselectsubset

Selects the given subset row in the table.

Syntax:

phasedbselectsubset ⟨row number⟩

Operands:
⟨row number⟩
The row number of the subset to select.

phasedbsetstorage

Records new storage options into the current Phase 3D Database. If the database already has clean output structures, then a job is started, if necessary, to modify the database ligand files (lists, structures, and sites) to match the new storage options. No action is taken if there is already a job active in the database, unless the job is killed first.

Syntax:

phasedbsetstorage *compressed*=yes | no *flexible*=yes | no

Options:

compressed
: Indicates whether or not the ligand files should be stored in compressed (gzip) form.
  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

flexible
: Indicates whether or not the database is to be configured for flexible searching. If true, then only a single conformation will be stored for each ligand in the database, and conformations and sites are generated only on-the-fly when database searching is done. If false, then multiple conformations can be generated and stored for the database ligands and sites files can be created and stored in the database.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

phasedbstepforward

Moves forward to the next Phase 3D Database step. Deletes any steps after the current step, then creates the next step, using the data from previous steps.

Syntax:

phasedbstepforward

phasedbstepgoto

Moves to an existing step in the current database.

Syntax:

phasedbstepgoto ⟨step name⟩

Operands:
⟨step name⟩
The name of the step to switch to.

phasedeletecustomfeature

Deletes the given custom feature.

Syntax:

phasedeletecustomfeature ⟨code⟩

Operands:
⟨code⟩
The code for the feature to delete.

phasedeleteexcludedvolumes

Deletes the selected excluded volumes.

Syntax:

phasedeleteexcludedvolumes

phasedeletehypothesis

Deletes the given hypothesis from the hypotheses table in the Score Hypotheses step.

Syntax:

phasedeletehypothesis ⟨ID⟩

Operands:
⟨ID⟩
The hypothesis to delete.

phasedeletehypothesisfrombuilder

Deletes the given hypothesis from the hypotheses table in the Edit Hypothesis and Find Matches panels.

Syntax:

phasedeletehypothesisfrombuilder ⟨row number⟩

Operands:
⟨row number⟩
The row number of the hypothesis to delete.

phasedeleteselectedligands

Deletes the selected ligands in the current Phase step.

Syntax:

phasedeleteselectedligands

phasedisplayproperty

Displays the given property from the table.

Syntax:

phasedisplayproperty ⟨property name⟩

Operands:
⟨property name⟩
The property to display.

phaseexcludetablerow

Excludes the given row in the first table in the step from the Workspace.

Syntax:

phaseexcludetablerow ⟨row⟩

Operands:
⟨row⟩
The row number to exclude in the Workspace.

phaseexportfeature

Exports the features to the given file name.

Syntax:

phaseexportfeature ⟨file name⟩

Operands:
⟨file name⟩
A path to a write the features file to.

phaseexporthypothesis

Exports the selected hypothesis from the hypotheses table in the Score Hypotheses step.

Syntax:

phaseexporthypothesis ⟨file name⟩

Operands:
⟨file name⟩
The file name to export the selected hypothesis to.

phaseexporthypothesisfrombuilder

Exports the selected hypothesis from the hypotheses table in the Edit Hypotheses panel.

Syntax:

phaseexporthypothesisfrombuilder ⟨file name⟩

Operands:
⟨file name⟩
The file name to export the selected hypothesis to.

phaseexportselectedalignments

Exports the selected alignments from the current Phase step to the project table.

Syntax:

phaseexportselectedalignments

phaseexportselectedligands

Exports the selected ligands from the current Phase step to the project table.

Syntax:

phaseexportselectedligands

phasefindmatches

Launches a Find Matches job.

Syntax:

phasefindmatches *align_weight*=⟨ x ⟩
apply_excluded_volumes=yes | no *database*=⟨ text ⟩
distance_tolerance=⟨ x ⟩ *fitness*=⟨ x ⟩ *hits_molecule*=⟨ n ⟩
hits_total=⟨ n ⟩ *jobinput*=selectedentries | 3ddatabase
matches=new | existing *minsites*=⟨ n ⟩ *useqsar*=yes | no
vector_weight=⟨ x ⟩ *volume_weight*=⟨ x ⟩ ⟨ job_id ⟩ | ⟨ job_name ⟩
| ⟨ job_name ⟩ ⟨ project_name ⟩

Options:

align_weight
The weight for the alignment score.

Valid values:	reals
Default value:	**1.2**
Minimum:	0.0001

apply_excluded_volumes
Indicates whether or not to use the excluded volumes information.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

database The source database for Phase Find Matches.

Valid values: text strings
Default value:

distance_tolerance
Valid values: reals
Default value: **2**

fitness The weight for the fitness score.
Valid values: reals
Default value: **1**
Minimum: 0.0

hits_molecule
The maximum number of hits per molecule to return.
Valid values: integers
Default value: **1**
Minimum: 1

hits_total The maximum number of hits to return.
Valid values: integers
Default value: **1000**
Minimum: 1

jobinput The source of job input of Phase Find Matches.
Valid values: selectedentries
3ddatabase
Default value: **3ddatabase**

matches Whether to search for new or existing matches.
Valid values: new
existing
Default value: **new**

minsites The minimum of sites to match.
Valid values: integers
Default value: **5**
Minimum: 3

useqsar The flag of using QSAR model or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

vector_weight
The weight for the vector score.
Valid values: reals
Default value: **1**
Minimum: 0.0

volume_weight
The weight for the volume score.

Valid values: reals
Default value: **1**
Minimum: 0.0

Operands:
⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩
The ID or name of the Find Matches job which is to be run.

phasefindpharmacophores

Sets the Find Common Pharmacophores options and launches the job.

Syntax:

phasefindpharmacophores *finalboxsize*=⟨x⟩ *maxtreedepth*=⟨n⟩ *minintersitedistance*=⟨x⟩

Options:

finalboxsize
The final box size in angstrong.

Valid values: reals
Default value: **1**

maxtreedepth
The maximum depth of the tree.

Valid values: integers
Default value: **4**

minintersitedistance
The minimum distance between the sites in angstrong.

Valid values: reals
Default value: **2**

phasegenerateconformers

Launches a Generate Conformers job for Phase.

Syntax:

phasegenerateconformers

phasehookimport

Sets up the Import panel and dialog for Phase to import ligands and conformers.

Syntax:

phasehookimport

phasehypothesiscreateexcludedvolume

Creates an excluded volume from the current centroid atoms for the currently selected hypothesis.

Syntax:

phasehypothesiscreateexcludedvolume

phasehypothesisdeleteexcludedvolumes

Deletes the selected excluded volumes for the currently selected hypothesis.

Syntax:

phasehypothesisdeleteexcludedvolumes

phasehypothesisselectevrow

Selects the given row in the excluded volumes table for the currently selected hypothesis.

Syntax:

phasehypothesisselectevrow ⟨row⟩

Operands:
⟨row⟩
The row number to select in the table.

phasehypothesisselectextendevrow

Extends the selection to this row in the excluded volumes table for the currently selected hypothesis.

Syntax:

phasehypothesisselectextendevrow ⟨row⟩

Operands:
⟨row⟩
The row number to extend the select to.

phasehypothesisselectonlyevrow

Selects only this row in the excluded volumes table for the currently selected hypothesis.

Syntax:

phasehypothesisselectonlyevrow ⟨row⟩

Operands:
⟨row⟩
The row number to select only in the table row.

phasehypothesissetexcludedvolumes

Sets the value for the given cell for the currently selected hypothesis.

Syntax:

phasehypothesissetexcludedvolumes *column*=⟨ n ⟩ *row*=⟨ n ⟩ ⟨ value ⟩

Options:

column The column number of the cell to change.
Valid values: integers
Default value: **1**

row The row number of the cell to change.
Valid values: integers
Default value: **1**

Operands:
⟨ value ⟩
The value for the given cell.

phasehypothesistoggleinclude

Toggles the given hypothesis into or out of the Workspace.

Syntax:

phasehypothesistoggleinclude ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to toggle inclusion state for.

phasehypothesisunselectevrow

Unselects the given row in the excluded volumes table for the currently selected hypothesis.

Syntax:

phasehypothesisunselectevrow ⟨row⟩

Operands:
⟨row⟩
The row number to unselect in the table.

phaseimportfeature

Imports the features from the given file name. This replaces the existing features.

Syntax:

phaseimportfeature ⟨file name⟩

Operands:
⟨file name⟩
A path to a valid features file.

phaseimporthypothesis

Imports a hypothesis from the selected file into the Find Matches and Edit Hypotheses panels.

Syntax:

phaseimporthypothesis *isphase*=yes | no ⟨file name⟩

Options:

isphase Indicates whether or not the hypothesis is a Phase hypothesis or an external hypothesis.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨file name⟩
The file name to import a hypothesis from.

phaseincludeextendtablerow

Extends the rows included in the workspace to include this one.

Syntax:

phaseincludeextendtablerow ⟨row⟩

Operands:
⟨row⟩
The row number to include in the Workspace.

phaseincludeonlytablerow

Includes only the given row in the first table in the step into the Workspace.

Syntax:

phaseincludeonlytablerow ⟨row⟩

Operands:
⟨row⟩
The row number to include in the Workspace.

phaseincludetablerow

Includes the given row in the first table in the step into the Workspace.

Syntax:

phaseincludetablerow ⟨row⟩

Operands:
⟨row⟩
The row number to include in the Workspace.

phaseinverttableselection

Inverts the row selection in the first table in the step.

Syntax:

phaseinverttableselection

phaseligprep

Defines settings for Phase LigPrep job.

Syntax:

phaseligprep *gen_stereo*=⟨ n ⟩ *ionization*=generate | neutralize | retain *ph*=⟨ x ⟩ *stereoisomers*=retain | determine | generate

Options:

gen_stereo Generate stereoisomers (maximum)

Valid values:	integers
Default value:	**32**
Minimum:	1

ionization How to generate the ionization states

Valid values:	generate neutralize retain
Default value:	**retain**

ph The ionization pH

Valid values:	reals
Default value:	**7**
Minimum:	0.0

stereoisomers
How to generate the stereoisomers

Valid values:	retain determine generate
Default value:	**retain**

phaselocatesites

Launches a Locate Sites job.

Syntax:

phaselocatesites

phasemarkerdump

Print out the current option values of the phase marker command.

Syntax:

phasemarkerdump

phasemarkersettings

Set graphical data of feature markers.

Syntax:

phasemarkersettings *acceptorarrowblue*=⟨ x ⟩
acceptorarrowgreen=⟨ x ⟩ *acceptorarrowred*=⟨ x ⟩
acceptorcornradius=⟨ x ⟩ *acceptorcylinderheight*=⟨ x ⟩
acceptorcylinderradius=⟨ x ⟩ *acceptorsiteblue*=⟨ x ⟩
acceptorsitegreen=⟨ x ⟩ *acceptorsiteradius*=⟨ x ⟩
acceptorsitered=⟨ x ⟩ *ambient*=⟨ x ⟩ *aromaticringradius*=⟨ x ⟩
aromaticringradius5=⟨ x ⟩ *aromaticsiteblue*=⟨ x ⟩
aromaticsitegreen=⟨ x ⟩ *aromaticsitered*=⟨ x ⟩
aromatictuberadius=⟨ x ⟩ *aromaticustepline*=⟨ n ⟩
aromaticustepsolid=⟨ n ⟩ *aromaticvstepline*=⟨ n ⟩
aromaticvstepsolid=⟨ n ⟩ *diffuse*=⟨ x ⟩ *donorarrowblue*=⟨ x ⟩
donorarrowgreen=⟨ x ⟩ *donorarrowred*=⟨ x ⟩
donorcornradius=⟨ x ⟩ *donorcylinderheight*=⟨ x ⟩
donorcylinderradius=⟨ x ⟩ *donorsiteblue*=⟨ x ⟩
donorsitegreen=⟨ x ⟩ *donorsiteradius*=⟨ x ⟩ *donorsitered*=⟨ x ⟩
drawstyle=solid | line *emission*=⟨ x ⟩ *featurexsiteblue*=⟨ x ⟩
featurexsitegreen=⟨ x ⟩ *featurexsiteradius*=⟨ x ⟩
featurexsitered=⟨ x ⟩ *featureysiteblue*=⟨ x ⟩
featureysitegreen=⟨ x ⟩ *featureysiteradius*=⟨ x ⟩
featureysitered=⟨ x ⟩ *featurezsiteblue*=⟨ x ⟩
featurezsitegreen=⟨ x ⟩ *featurezsiteradius*=⟨ x ⟩
featurezsitered=⟨ x ⟩ *hydrophobicsiteblue*=⟨ x ⟩
hydrophobicsitegreen=⟨ x ⟩ *hydrophobicsiteradius*=⟨ x ⟩
hydrophobicsitered=⟨ x ⟩ *linewidth*=⟨ n ⟩ *negativesiteblue*=⟨ x ⟩
negativesitegreen=⟨ x ⟩ *negativesiteradius*=⟨ x ⟩
negativesitered=⟨ x ⟩ *positivesiteblue*=⟨ x ⟩ *positivesitegreen*=⟨ x ⟩
positivesiteradius=⟨ x ⟩ *positivesitered*=⟨ x ⟩ *shininess*=⟨ x ⟩
sliceline=⟨ n ⟩ *slicesolid*=⟨ n ⟩ *specular*=⟨ x ⟩ *stackline*=⟨ n ⟩
stacksolid=⟨ n ⟩ *transparency*=⟨ x ⟩

Options:

acceptorarrowblue

The arrow color blue component of acceptor markers.

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0
Maximum:	1.0

acceptorarrowgreen

The arrow color green component of acceptor markers.

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0
Maximum:	1.0

acceptorarrowred

The arrow color red component of acceptor markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

acceptorcornradius

The radius of arrow corn of acceptor markers.

Valid values: reals
Default value: **0.35**
Minimum: 0.0

acceptorcylinderheight

The height of arrow cylinder of acceptor markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0
Maximum: 1.0

acceptorcylinderradius

The radius of arrow cylinder of acceptor markers.

Valid values: reals
Default value: **0.15**
Minimum: 0.0

acceptorsiteblue

The site color blue component of acceptor markers.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

acceptorsitegreen

The site color green component of acceptor markers.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

acceptorsiteradius

The radius of acceptor site markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0

acceptorsitered
The site color red component of acceptor markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

ambient Set material property - ambient, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

aromaticringradius
The radius of the ring of the ring torus, for the ring constructed by 6 or more atoms.

Valid values: reals
Default value: **0.5**
Minimum: 0.0

aromaticringradius5
The radius of the ring of the ring torus, for the ring constructed by 5 or less atoms.

Valid values: reals
Default value: **0.4**
Minimum: 0.0

aromaticsiteblue
The site color blue component of aromatic markers.

Valid values: reals
Default value: **0.2**
Minimum: 0.0
Maximum: 1.0

aromaticsitegreen
The site color green component of aromatic markers.

Valid values: reals
Default value: **0.51**
Minimum: 0.0
Maximum: 1.0

aromaticsitered
The site color red component of aromatic markers.

Valid values: reals
Default value: **0.92**

Minimum: 0.0
Maximum: 1.0

aromatictuberadius
The radius of the tube of the ring torus.

Valid values: reals
Default value: **0.15**
Minimum: 0.0

aromaticustepline
The U step domain tolerance of frame torus.

Valid values: integers
Default value: **5**
Minimum: 2

aromaticustepsolid
The U step domain tolerance of solid torus.

Valid values: integers
Default value: **10**
Minimum: 2

aromaticvstepline
The V step domain tolerance of frame torus.

Valid values: integers
Default value: **3**
Minimum: 2

aromaticvstepsolid
The V step domain tolerance of solid torus.

Valid values: integers
Default value: **5**
Minimum: 2

diffuse Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.4**
Minimum: 0.0
Maximum: 1.0

donorarrowblue
The arrow color blue component of donor markers.

Valid values: reals
Default value: **0.9**
Minimum: 0.0
Maximum: 1.0

donorarrowgreen

The arrow color green component of donor markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0
Maximum: 1.0

donorarrowred

The arrow color red component of donor markers.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

donorcornradius

The radius of arrow corn of donor markers.

Valid values: reals
Default value: **0.35**
Minimum: 0.0

donorcylinderheight

The height of arrow cylinder of donor markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0
Maximum: 1.0

donorcylinderradius

The radius of arrow cylinder of donor markers.

Valid values: reals
Default value: **0.15**
Minimum: 0.0

donorsiteblue

The site color blue component of donor markers.

Valid values: reals
Default value: **0.9**
Minimum: 0.0
Maximum: 1.0

donorsitegreen

The site color green component of donor markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0
Maximum: 1.0

donorsiteradius
The radius of donor site markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0

donorsitered
The site color red component of donor markers.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

drawstyle The styles of rendering feature markers, they are: 1 - solid, and 2 - lines. Default is solid.

Valid values: solid
line
Default value: **solid**

emission Set material property - emission, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.05**
Minimum: 0.0
Maximum: 1.0

featurexsiteblue
The site color blue component of featurex markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

featurexsitegreen
The site color green component of featurex markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

featurexsiteradius
The radius of featurex site markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0

featurexsitered
The site color red component of featurex markers.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

featureysiteblue
The site color blue component of featurey markers.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

featureysitegreen
The site color green component of featurey markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

featureysiteradius
The radius of featurey site markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0

featureysitered
The site color red component of featurey markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

featurezsiteblue
The site color blue component of featurez markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

featurezsitegreen
The site color green component of featurez markers.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

featurezsiteradius
The radius of featurez site markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0

featurezsitered
The site color red component of featurez markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

hydrophobicsiteblue
The site color blue component of hydrophobic markers.

Valid values: reals
Default value: **0.3**
Minimum: 0.0
Maximum: 1.0

hydrophobicsitegreen
The site color green component of hydrophobic markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

hydrophobicsiteradius
The radius of hydrophobic site markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0

hydrophobicsitered
The site color red component of hydrophobic markers.

Valid values: reals
Default value: **0.3**
Minimum: 0.0
Maximum: 1.0

linewidth Set the width of lines in drawing sphere.

Valid values: integers
Default value: **2**
Minimum: 1

negativesiteblue
The site color blue component of negative markers.

Valid values: reals
Default value: **0.3**

Minimum: 0.0
Maximum: 1.0

negativesitegreen
The site color green component of negative markers.

Valid values: reals
Default value: **0.3**
Minimum: 0.0
Maximum: 1.0

negativesiteradius
The radius of negative site markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0

negativesitered
The site color red component of negative markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

positivesiteblue
The site color blue component of positive markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

positivesitegreen
The site color green component of positive markers.

Valid values: reals
Default value: **0.6**
Minimum: 0.0
Maximum: 1.0

positivesiteradius
The radius of positive site markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0

positivesitered
The site color red component of positive markers.

Valid values: reals
Default value: **0.5**

Minimum: 0.0
Maximum: 1.0

shininess Set material property - shininess, for front face.

Valid values: reals
Default value: **80**
Minimum: 0.0
Maximum: 128.0

sliceline Set the slices of drawing line sphere.

Valid values: integers
Default value: **10**
Minimum: 2

slicesolid Set the slices of drawing solid sphere.

Valid values: integers
Default value: **36**
Minimum: 2

specular Set material property - specular, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

stackline Set the stacks of drawing line sphere.

Valid values: integers
Default value: **8**
Minimum: 2

stacksolid Set the stacks of drawing solid sphere.

Valid values: integers
Default value: **18**
Minimum: 2

transparency
The transparency of rendering feature markers.

Valid values: reals
Default value: **20**
Minimum: 0.0
Maximum: 100.0

phasemarkfeature

Marks the given feature in the Workspace.

Syntax:

phasemarkfeature ⟨feature⟩

Operands:
⟨feature⟩
A single letter (A-Z) indicating the feature to mark.

phaseoptions

Sets the options for Phase.

Syntax:

phaseoptions *random_seed*=⟨n⟩ *separate_stereoisomers*=yes | no

Options:

random_seed
The seed used to compute the random training / test set in the Build QSAR step. Zero means to use a completely random seed–any other value is used explicitly.

Valid values: integers
Default value: **0**
Minimum: 0

separate_stereoisomers
Whether to separate stereoisomers or combine them into a single confset.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

phasepatterndelete

Deletes the given pattern.

Syntax:

phasepatterndelete *pattern*=⟨text⟩ ⟨feature⟩

Options:

pattern The SMARTS pattern to remove.
Valid values: text strings
Default value:

Operands:
⟨feature⟩
The feature to remove the pattern from.

phasepatternedit

Updates the given pattern with the new data.

Syntax:

phasepatternedit *geometry*=group | point | vector *group*=⟨text⟩ *group_all*=yes | no *pattern*=⟨text⟩ *point*=⟨n⟩ *vector*=⟨n⟩ ⟨feature⟩

Options:

geometry The type of geometry for this pattern.
Valid values: group
point
vector
Default value: **group**

group The atom numbers for the pattern if it is group geometry.
Valid values: text strings
Default value:

group_all Set to true if the group should include all atom numbers.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

pattern The SMARTS pattern to update.
Valid values: text strings
Default value:

point The atom number for the pattern if it is point geometry.
Valid values: integers

Default value: **1**
Minimum: 1

vector The atom number for the pattern if it is vector geometry.

Valid values: integers
Default value: **1**
Minimum: 1

Operands:

⟨ feature ⟩

The feature to which the pattern is associated with.

phasepatternmovedown

Moves the given pattern down lower in the list for the feature.

Syntax:

phasepatternmovedown *pattern*=⟨ text ⟩ ⟨ feature ⟩

Options:

pattern The SMARTS pattern to move.

Valid values: text strings
Default value:

Operands:

⟨ feature ⟩

The feature which contains the pattern.

phasepatternmoveup

Moves the given pattern up higher in the list for the feature.

Syntax:

phasepatternmoveup *pattern*=⟨text⟩ ⟨feature⟩

Options:

pattern The SMARTS pattern to move.
Valid values: text strings
Default value:

Operands:
⟨feature⟩
The feature which contains the pattern.

phasepatternnew

Adds the given pattern.

Syntax:

phasepatternnew *geometry*=group | point | vector *group*=⟨text⟩ *group_all*=yes | no *pattern*=⟨text⟩ *point*=⟨n⟩ *vector*=⟨n⟩ ⟨feature⟩

Options:

geometry The type of geometry for this pattern.
Valid values: group
point
vector
Default value: **group**

group The atom numbers for the pattern if it is group geometry.
Valid values: text strings
Default value:

group_all Set to true if the group should include all atom numbers.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

pattern The SMARTS pattern to add.
Valid values: text strings
Default value:

point The atom number for the pattern if it is point geometry.
Valid values: integers

Default value: **1**
Minimum: 1

vector The atom number for the pattern if it is vector geometry.
Valid values: integers
Default value: **1**
Minimum: 1

Operands:
⟨ feature ⟩
The feature to add the pattern to.

phasepatternsetoptions

Sets the options for the given pattern.

Syntax:

phasepatternsetoptions *exclude*=yes | no *ignore*=yes | no *mark*=yes | no *pattern*=⟨ text ⟩ ⟨ feature ⟩

Options:

exclude Indicates whether or not to exclude the pattern from the feature.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

ignore Indicates whether or not to ignore the pattern in the feature.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

mark Indicates whether or not to mark the pattern in the Workspace.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

pattern The SMARTS pattern to update.
Valid values: text strings
Default value:

Operands:
⟨ feature ⟩
The feature which contains the pattern.

phasepreview

Launches a Preview job.

Syntax:

phasepreview

phaseqsarsearch

Transfers the current hypothesis from the Build QSAR step to the Find Matches panel. Also opens the Find Matches panel.

Syntax:

phaseqsarsearch

phaserandomtraining

Sets the random training set.

Syntax:

phaserandomtraining

phaserandomtrainingoptions

Holds the options for the random training set.

Syntax:

phaserandomtrainingoptions *percentage*=⟨ n ⟩

Options:

percentage The percentage of ligands to use as a training set.
Valid values: integers
Default value: **50**

Minimum: 1
Maximum: 100

phaserefreshfrequencytable

Restores the default set of min/max values to the feature frequencies table in the Find Common Pharamcophores step.

Syntax:

phaserefreshfrequencytable

phaseremoveligands

Removes the given ligands from the table.

Syntax:

phaseremoveligands ⟨ESL⟩

Operands:
⟨ESL⟩
The entries to remove as ligands.

phaserenameligand

Changes the name for the given ligand.

Syntax:

phaserenameligand *row*=⟨n⟩ ⟨new name⟩

Options:

row The row number of the ligand to rename.
Valid values: integers
Default value: **1**
Minimum: 1

Operands:
⟨ new name ⟩
The new name for the given ligand.

phaserescorehypotheses

Rescores all the hypotheses in the hypotheses table in the Score Hypotheses step.

Syntax:

phaserescorehypotheses

phaserescoreweighting

Sets the weighting factors for rescoring hypotheses in the Score Hypotheses step.

Syntax:

phaserescoreweighting *selectivity*=⟨ x ⟩ *site*=⟨ x ⟩ *vector*=⟨ x ⟩ *volume*=⟨ x ⟩

Options:

selectivity The selectivity scoring factor.
Valid values: reals
Default value: **1**

site The aligned sites scoring factor.
Valid values: reals
Default value: **1**

vector The vector scoring factor.
Valid values: reals
Default value: **1**

volume The volume scoring factor.
Valid values: reals
Default value: **1**

phaseresetfeatures

Resets the features to the installation defaults.

Syntax:

phaseresetfeatures

phaseruncreate

Creates the run with the given name.

Syntax:

phaseruncreate ⟨run name⟩

Operands:
⟨run name⟩
The name of the new run to create.

phaserundelete

Deletes the current run from the project.

Syntax:

phaserundelete

phaserunopen

Opens the run with the given name.

Syntax:

phaserunopen ⟨run name⟩

Operands:

⟨run name⟩
The name of the run to open.

phaserunrename

Changes the current run's name to the given name.

Syntax:

phaserunrename ⟨run name⟩

Operands:
⟨run name⟩
The name to change the current run's name to.

phaserunsaveas

Saves a copy of the current run under the given name.

Syntax:

phaserunsaveas ⟨run name⟩

Operands:
⟨run name⟩
The name of the run to save as.

phaserunsetseed

Sets the random seed of the current run based on the current Phase options.

Syntax:

phaserunsetseed

phaserunsetstereo

Sets the stereoisomer behavior of the current run based on the current Phase options.

Syntax:

phaserunsetstereo

phasescorehypotheses

Launches a Score Hypotheses job.

Syntax:

phasescorehypotheses *aligncutoff*=⟨ x ⟩ *matchreward*=⟨ x ⟩ *maxboxes*=⟨ n ⟩ *minboxes*=⟨ n ⟩ *selectivitybasedalignment*=yes | no *selectivityweight*=⟨ x ⟩ *sitetoppercentage*=⟨ n ⟩ *siteweight*=⟨ x ⟩ *vectorlowercutoff*=⟨ x ⟩ *vectorweight*=⟨ x ⟩ *volumebasedalignment*=yes | no *volumeweight*=⟨ x ⟩

Options:

aligncutoff

In case of score by site-based alignment, keep the hypotheses with RMSD values below the value specified by this option. The valid range is 0.0001 to infinity

Valid values: reals
Default value: **1.2**
Minimum: 0.0001

matchreward

For total scoring, increase the score by this value, raised to the power of the number of matched ligands. The valid range is 1.0 to infinity, but values only slightly above 1.0 might generate huge values for a large number of matched ligands.

Valid values: reals
Default value: **1**

Minimum: 1.0

maxboxes In case of score by site-based alignment, keep at most the number specified by this option. The valid range is 1 to infinity

Valid values: integers
Default value: **50**
Minimum: 1

minboxes In case of score by site-based alignment, keep at least the number specified by this option. The valid range is 1 to infinity

Valid values: integers
Default value: **10**
Minimum: 1

selectivitybasedalignment
For score by selectivity-based alignment, this option is true otherwise false.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

selectivityweight
For calculating the Survival score, the weighting factor of the selectivity score. The valid range is 0.0 to 1.0

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

sitetoppercentage
In case of score by site-based alignment, keep the hypotheses in the percentage specified by this option. The valid range is 0 to 100

Valid values: integers
Default value: **10**
Minimum: 0
Maximum: 100

siteweight For calculating the Survival score, the weighting factor given for the site score. The valid range is 0.0 to 1.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

vectorlowercutoff
In case of score by vector-based alignment, keep the hypotheses that score above the value specified by this option. The valid range is 0.0 to 1.0

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

vectorweight
For calculating the Survival score, the weighting factor given for the vector score. The valid range is 0.0 to 1.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

volumebasedalignment
For score by volume-based alignment, this option is true otherwise false.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

volumeweight
For calculating the Survival score, the weighting factor given for the volume score. The valid range is 0.0 to 1.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

phasescoreselecthypothesis

Selects the given hypothesis.

Syntax:

phasescoreselecthypothesis *alignall*=yes | no ⟨ID⟩

Options:

alignall This option is used to control the behavior when filling in hypothesis alignments table. If this is "true" then alignments are generated for all ligands, including inactives, that match the hypothesis on at least 3 sites. If this is "false" then the table is filled in with only the ligands from the active set that match all of the hypothesis sites.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

Operands:
⟨ ID ⟩
The ID of the hypothesis to select. This will populate the Alignment table. This function is single select. If the operand is blank, the alignments table will be filled in for the currently selected hypothesis, if any.

phasescoreweighting

Sets the weighting factors for scoring hypotheses in the Score Hypotheses step.

Syntax:

phasescoreweighting *selectivity*=⟨ x ⟩ *site*=⟨ x ⟩ *vector*=⟨ x ⟩ *volume*=⟨ x ⟩

Options:

selectivity The selectivity scoring factor.
Valid values: reals
Default value: **1**

site The aligned sites scoring factor.
Valid values: reals
Default value: **1**

vector The vector scoring factor.
Valid values: reals
Default value: **1**

volume The volume scoring factor.
Valid values: reals
Default value: **1**

phaseselectalltablerows

Selects all rows in the first table in the step.

Syntax:

phaseselectalltablerows

phaseselectevrow

Selects the given row in the excluded volumes table.

Syntax:

phaseselectevrow ⟨row⟩

Operands:
⟨row⟩
The row number to select in the table.

phaseselectextendevrow

Extends the selection to this row in the excluded volumes table.

Syntax:

phaseselectextendevrow ⟨row⟩

Operands:
⟨row⟩
The row number to extend the select to.

phaseselectextendtablerow

Extends the selection to this row in the table.

Syntax:

phaseselectextendtablerow ⟨row⟩

Operands:
⟨row⟩
The row number to extend the select to.

phaseselectonlyevrow

Selects only this row in the excluded volumes table.

Syntax:

phaseselectonlyevrow ⟨row⟩

Operands:
⟨row⟩
The row number to select only in the table row.

phaseselectonlytablerow

Selects only this row in the table.

Syntax:

phaseselectonlytablerow ⟨row⟩

Operands:
⟨row⟩
The row number to select only in the table row.

phaseselecttablerow

Selects the given row in the first table in the step.

Syntax:

phaseselecttablerow ⟨row⟩

Operands:
⟨row⟩
The row number to select in the table.

phasesetactivity

Sets the activity for the given row in the current step.

Syntax:

phasesetactivity *activity*=⟨x⟩ ⟨row⟩

Options:

activity The activity value for the given ligand.
Valid values: reals
Default value: **1**

Operands:
⟨row⟩
The row number of the ligand to set the activity for.

phasesetactivityproperty

Sets the activity property.

Syntax:

phasesetactivityproperty ⟨property⟩

Operands:
⟨property⟩
The property to get the activity values from.

phasesetexcludedvolumes

Sets the value for the given cell.

Syntax:

phasesetexcludedvolumes *column*=⟨ n ⟩ *row*=⟨ n ⟩ ⟨ value ⟩

Options:

column The column number of the cell to change.

Valid values: integers
Default value: **1**

row The row number of the cell to change.

Valid values: integers
Default value: **1**

Operands:
⟨ value ⟩
The value for the given cell.

phasesetfrequency

Sets the minimum or maximum frequency for a given feature.

Syntax:

phasesetfrequency *column*=⟨ n ⟩ *value*=⟨ n ⟩ ⟨ row ⟩

Options:

column The table column to set.

Valid values: integers
Default value: **2**
Minimum: 1

value The minimum or maximum frequency.

Valid values: integers
Default value: **0**
Minimum: 0

Operands:
⟨ row ⟩
The row number of the min / max value to set.

phasesettrainingrows

Toggles the Training Set property on or off for the given row in the Alignments table under Build QSAR step.

Syntax:

phasesettrainingrows *value*=training | test | none ⟨ rows ⟩

Options:

value The value of the training/test column in the alignment table under Build QSAR Model step.

Valid values:	training test none
Default value:	**training**

Operands:
⟨ rows ⟩
The row numbers to toggle, two row number should be seprated by , or if user wants to specify a range then it can be given like 1-5 seperated by a - . eg 1,2,5 or 1,4-7 .

phasesiteoptions

Sets the site options for a Find Pharmacophores job.

Syntax:

phasesiteoptions *match*=all | minimum *minimum*=⟨ n ⟩
numsites=⟨ n ⟩

Options:

match Indicates whether to match against all or a minimum set of active ligands.

Valid values: all
minimum
Default value: **all**

minimum The minimum number of active ligands which a pattern has to match against.

Valid values: integers
Default value: **1**
Minimum: 1

numsites The number of sites to match.

Valid values: integers
Default value: **5**
Minimum: 3
Maximum: 7

phasesorttable

Resort the given Phase table based on the data in the specified column

Syntax:

phasesorttable *table*=⟨ n ⟩ ⟨ column_name ⟩

Options:

table The table to set.

Valid values: integers
Default value: **10**

Operands:
⟨ column_name ⟩
The name of the column to be sorted.

phasestepforward

Moves forward to the next Phase step. Deletes any steps after the current step, then creates the next step, using the data from previous steps.

Syntax:

phasestepforward

phasestepgoto

Moves to an existing step in the current project.

Syntax:

phasestepgoto ⟨step name⟩

Operands:
⟨step name⟩
The name of the step to switch to.

phasetoggleactivetablerow

Toggles the Active property on or off for the given row in the first table in the step.

Syntax:

phasetoggleactivetablerow ⟨row⟩

Operands:
⟨row⟩
The row number to toggle.

phasetogglecentroidatom

Adds or removes the given atom from the current centroid list for excluded volumes.

Syntax:

phasetogglecentroidatom ⟨atom index⟩

Operands:
⟨atom index⟩
The index of the atom to toggle.

phasetoggletrainingrow

Toggles the Training Set property on or off for the given row in the Build QSAR step.

Syntax:

phasetoggletrainingrow ⟨row⟩

Operands:
⟨row⟩
The row number to toggle.

phaseundisplayproperty

Undisplays the given property from the table.

Syntax:

phaseundisplayproperty ⟨property name⟩

Operands:
⟨property name⟩
The property to undisplay.

phaseunmarkfeature

Unmarks the given feature in the Workspace.

Syntax:

phaseunmarkfeature ⟨feature⟩

Operands:
⟨feature⟩
A single letter (A-Z) indicating the feature to unmark.

phaseunselectevrow

Unselects the given row in the excluded volumes table.

Syntax:

phaseunselectevrow ⟨row⟩

Operands:
⟨row⟩
The row number to unselect in the table.

phaseunselecttablerow

Unselects the given row in the first table in the step.

Syntax:

phaseunselecttablerow ⟨row⟩

Operands:
⟨row⟩
The row number to unselect in the table.

picksize

This command allows the user to choose the size of the pick box that is used for picking atoms/bonds/residues etc.

Syntax:

picksize ⟨size⟩

Operands:
⟨size⟩
This operand actually defines the size of the pick box. Allowed sizes are 7X7, 10X10 and 15X15 which are specified at command line as 7, 10 & 15 respectively.

place

Place the current fragment on screen at the <x>, <y> and <z> positions given by the operands.

Syntax:

place ⟨x⟩ ⟨y⟩ ⟨z⟩

Operands:
⟨x⟩ ⟨y⟩ ⟨z⟩
The operands are three real numbers which are the x, y and z coordinates where the new fragment is to be placed.

plotxyarrangecolumn

Puts all of the plots in a column.

Syntax:

plotxyarrangecolumn

plotxyarrangerow

Displays the plots in a row.

Syntax:

plotxyarrangerow

plotxyarrangetiled

Tiles all of the displayed plots.

Syntax:

plotxyarrangetiled

plotxyaxis

Creates or modifies an XY axis.

Syntax:

plotxyaxis *axis*=x | y *maximum*=⟨ text ⟩ *minimum*=⟨ text ⟩ *nummarkers*=⟨ n ⟩ *plot*=⟨ text ⟩ ⟨ title ⟩

Options:

axis This option indicates whether this axis is an X axis or a Y axis.
Valid values: x
y
Default value: **x**

maximum This option sets the maximum value for the axis.
Valid values: text strings
Default value:

minimum This option sets the minimum value for the axis.
Valid values: text strings
Default value:

nummarkers
This option sets the number of scale markers for the axis.
Valid values: integers
Default value: **10**
Minimum: 0

plot This option is the name of the plot containing the axis.
Valid values: text strings
Default value:

Operands:
⟨ title ⟩
The name of the axis to create or modify.

plotxyaxisautorange

Toggles autorange for the given axis on or off.

Syntax:

plotxyaxisautorange *autorange*=yes | no *axis*=x | y
plot=⟨ text ⟩ ⟨ axisname ⟩

Options:

autorange This option indicates whether or not to enable autorange for this axis.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

axis This option indicates whether this axis is an X axis or a Y axis.
Valid values: x
y
Default value: **x**

plot This option is the name of the plot containing the axis.
Valid values: text strings
Default value:

Operands:
⟨ axisname ⟩
The name of the axis to set autorange for.

plotxyaxisdelete

Deletes an XY plot axis.

Syntax:

plotxyaxisdelete *axis*=x | y *plot*=⟨text⟩ ⟨axisname⟩

Options:

axis This option indicates whether this axis is an X axis or a Y axis.
Valid values: x
y
Default value: **x**

plot This option is the name of the plot containing the axis.
Valid values: text strings
Default value:

Operands:
⟨axisname⟩
The name of the axis to delete.

plotxyaxisdisplay

Toggles display of the given axis on or off.

Syntax:

plotxyaxisdisplay *autorange*=yes | no *axis*=x | y *plot*=⟨text⟩ ⟨axisname⟩

Options:

autorange This option indicates whether to display or undisplay the given axis.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

axis This option indicates whether this axis is an X axis or a Y axis.
Valid values: x
y
Default value: **x**

plot This option is the name of the plot containing the axis.
Valid values: text strings
Default value:

Operands:
⟨ axisname ⟩
The name of the axis to display or undisplay.

plotxyaxisrename

Renames an existing axis.

Syntax:

plotxyaxisrename *axis*=x | y *newname*=⟨ text ⟩ *plot*=⟨ text ⟩
⟨ title ⟩

Options:

axis This option indicates whether this axis is an X axis or a Y axis.
Valid values: x
y
Default value: **x**

newname This option is the new name for the axis.
Valid values: text strings
Default value:

plot This option is the name of the plot containing the axis.
Valid values: text strings
Default value:

Operands:
⟨ title ⟩
The name of the axis to rename.

plotxycaption

Sets the caption for the given plot.

Syntax:

plotxycaption ⟨plotname⟩ ⟨caption⟩

Operands:
⟨plotname⟩ ⟨caption⟩
The name of the plot to operate on. The new caption for the plot.

plotxycaptionposition

Sets the position of the caption in the given plot.

Syntax:

plotxycaptionposition ⟨plotname⟩ top|bottom

Operands:
⟨plotname⟩ top|bottom
The name of the plot to operate on. top to display the caption at the top of the plot or bottom to display it at the bottom.

plotxycopy

Copies the selected plots.

Syntax:

plotxycopy

plotxydelete

Deletes the selected plots.

Syntax:

plotxydelete

plotxydisplay

Displays the selected plots in addition to any currently displayed plots.

Syntax:

plotxydisplay

plotxydisplaycaption

Shows or hides the caption for the given plot.

Syntax:

plotxydisplaycaption ⟨plotname⟩ yes|no

Operands:
⟨plotname⟩ yes|no
The name of the plot to operate on. yes to display the caption or no to not display the caption.

plotxydisplayincluded

Sets the Display Included Markers state for the given plot.

Syntax:

plotxydisplayincluded ⟨plotname⟩ yes|no

Operands:
⟨plotname⟩ yes|no
The name of the plot to operate on. yes to display included markers or no to not display them.

plotxydisplaylegend

Displays or hides the legend for the given plot.

Syntax:

plotxydisplaylegend ⟨plotname⟩ yes|no

Operands:
⟨plotname⟩ yes|no
The name of the plot to operate on. yes to display the legend or no to not display it.

plotxydisplayname

Shows or hides the name in the caption for the given plot.

Syntax:

plotxydisplayname ⟨plotname⟩ yes|no

Operands:
⟨plotname⟩ yes|no
The name of the plot to operate on. yes to display the plot name in the caption or no to not display the name in the caption.

plotxydisplayonly

Displays only the selected plots.

Syntax:

plotxydisplayonly

plotxydisplaypointlabels

Displays or hides the point labels for the given plot.

Syntax:

plotxydisplaypointlabels ⟨plotname⟩ yes|no

Operands:

⟨plotname⟩ yes|no

The name of the plot to operate on. yes to display the point labels or no to not display them.

plotxydisplayselected

Displays or hides the selected entry markers for the given plot.

Syntax:

plotxydisplayselected ⟨plotname⟩ yes|no

Operands:

⟨plotname⟩ yes|no

The name of the plot to operate on. yes to display the selected entry markers or no to not display them.

plotxyhidesidebar

Hides the plotxy side bar.

Syntax:

plotxyhidesidebar

plotxyhidetoolbar

Hides the plotxy tool bar.

Syntax:

plotxyhidetoolbar

plotxylabel

Turns on labels for the given data point.

Syntax:

plotxylabel *entryname*=yes | no *plot*=⟨text⟩ *series*=⟨text⟩ *xaxis*=yes | no *yaxis*=yes | no ⟨entry⟩

Options:

entryname
: If this is set to true then the entry name is displayed as part of the label.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

plot
: This option is the name of the plot containing the series and entry.

Valid values: text strings
Default value:

series
: This option is the name of the series containing the entry.

Valid values: text strings
Default value:

xaxis
: If this is set to true then the X-axis property is displayed as part of the label.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

yaxis
: If this is set to true then the Y-axis property is displayed as part of the label.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:

⟨entry⟩

The name of the entry to label.

plotxymove

Moves the selected plots.

Syntax:

plotxymove ⟨location⟩

Operands:
⟨location⟩
The location to move the selected plots to. If the location is 0, then the plots are moved to the beginning of the plot list. Otherwise, the plots are moved after the given plot number in the list.

plotxynew

Creates a new XY plot

Syntax:

plotxynew ⟨plotname⟩

Operands:
⟨plotname⟩
The name of the plot to create.

plotxypan

Moves the selected plots.

Syntax:

plotxypan ⟨x⟩ ⟨y⟩

Operands:
⟨x⟩ ⟨y⟩
This is a percentage of the full range to offset by. This is a percentage of the full range to offset by.

plotxypanplot

Moves the specified plot.

Syntax:

plotxypanplot ⟨plot⟩ ⟨x⟩ ⟨y⟩

Operands:
⟨plot⟩ ⟨x⟩ ⟨y⟩
This is the name of the plot to pan. This is a percentage of the full range to offset by. This is a percentage of the full range to offset by.

plotxyrename

Renames the given plot to the new name.

Syntax:

plotxyrename ⟨plotname⟩ ⟨newname⟩

Operands:
⟨plotname⟩ ⟨newname⟩
The name of the plot to operate on. The new name for the plot.

plotxyresetview

Resets the zoom and pan for the selected plots.

Syntax:

plotxyresetview

plotxysaveimage

Capture the current XY plotting window and save it to an image file.

Syntax:

plotxysaveimage *format*=tiff | jpeg ⟨file_name⟩

Options:

format Specifies the format of the saved image.

Valid values: tiff
jpeg
Default value: **tiff**

Operands:
⟨file_name⟩
The file where the image will be saved.

plotxyselect

Selects the given plots.

Syntax:

plotxyselect ⟨plotname⟩

Operands:
⟨plotname⟩
Names of the plots to select, or all, or displayed.

plotxyseries

Creates or modifies an XY data series.

Syntax:

plotxyseries *color*=black | red | green | blue | purple | orange | blue_green | light_green | red_purple | yellow | cyan *marker*=filled_square | square | filled_circle | circle | cross | point | diamond | filled_diamond | none *plot*=⟨ text ⟩ *style*=none | solid | dash *unityaspect*=yes | no *width*=⟨ n ⟩ *xaxis*=⟨ text ⟩ *xbuckets*=⟨ n ⟩ *xproperty*=⟨ text ⟩ *yaxis*=⟨ text ⟩ *ybuckets*=⟨ n ⟩ *yproperty*=⟨ text ⟩ ⟨ seriesname ⟩

Options:

color The color of the symbols and lines used on the plot for this data series.

Valid values: black, red, green, blue, purple, orange, blue_green, light_green, red_purple, yellow, cyan

Default value: **black**

marker The marker for this data series.

Valid values: filled_square, square, filled_circle, circle, cross, point, diamond, filled_diamond, none

Default value: **filled_square**

plot This option is the name of the plot containing the series.

Valid values: text strings

Default value:

style The line style for this data series.

Valid values: none, solid, dash

Default value: **none**

unityaspect
Settings this option to yes causes the plot to display with an equal increment on each axis mapping to the same distance on the screen.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

width This option sets the width of the line for this data series.
Valid values: integers
Default value: **1**
Minimum: 1

xaxis This option selects an existing X-axis.
Valid values: text strings
Default value:

xbuckets This option sets the number of buckets for the X-axis.
Valid values: integers
Default value: **100**
Minimum: 0

xproperty This option sets the name of the property for the X-axis.
Valid values: text strings
Default value:

yaxis This option selects an existing Y-axis.
Valid values: text strings
Default value:

ybuckets This option sets the number of buckets for the Y-axis.
Valid values: integers
Default value: **100**
Minimum: 0

yproperty This option sets the name of the property for the Y-axis.
Valid values: text strings
Default value:

Operands:
⟨seriesname⟩
The name of the series to create or modify.

plotxyseriesdelete

Deletes the given XY data series.

Syntax:

plotxyseriesdelete *plot*=⟨text⟩ ⟨seriesname⟩

Options:

plot This option is the name of the plot containing the series.
Valid values: text strings
Default value:

Operands:
⟨seriesname⟩
The name of the series to delete.

plotxyseriesdisplay

Displays or undisplays the given series.

Syntax:

plotxyseriesdisplay ⟨plotname⟩ ⟨series⟩ yes|no

Operands:
⟨plotname⟩ ⟨series⟩ yes|no
The name of the plot to operate on. The name of the series. yes to display the given data series or no to not display it.

plotxyseriesrename

Renames an existing series.

Syntax:

plotxyseriesrename *newname*=⟨text⟩ *plot*=⟨text⟩ ⟨title⟩

Options:

newname This option is the new name for the series.
Valid values: text strings
Default value:

plot This option is the name of the plot containing the series.
Valid values: text strings
Default value:

Operands:
⟨ title ⟩
The name of the series to rename.

plotxyseriesselect

Selects only the entries corresponding to the given series.

Syntax:

plotxyseriesselect *plot*=⟨ text ⟩ ⟨ seriesname ⟩

Options:

plot This option is the name of the plot containing the series.
Valid values: text strings
Default value:

Operands:
⟨ seriesname ⟩
The name of the series to select.

plotxyseriesselectadd

Adds the entries corresponding to the given series to the current selection.

Syntax:

plotxyseriesselectadd *plot*=⟨ text ⟩ ⟨ seriesname ⟩

Options:

plot This option is the name of the plot containing the series.
Valid values: text strings
Default value:

Operands:
⟨seriesname⟩
The name of the series to select.

plotxyshowsidebar

Displays the plotxy side bar.

Syntax:

plotxyshowsidebar

plotxyshowtoolbar

Displays the plotxy tool bar.

Syntax:

plotxyshowtoolbar

plotxytoggledisplay

Toggles the display of the plot on or off.

Syntax:

plotxytoggledisplay ⟨plotname⟩

Operands:
⟨plotname⟩
The name of the plot to toggle the display of.

plotxyundisplay

Undisplays the selected plots.

Syntax:

plotxyundisplay

plotxyunlabel

Turns off the label for the given data point.

Syntax:

plotxyunlabel *plot*=⟨text⟩ *series*=⟨text⟩ ⟨entry⟩

Options:

plot This option is the name of the plot containing the series and entry.
Valid values: text strings
Default value:

series This option is the name of the series containing the entry.
Valid values: text strings
Default value:

Operands:
⟨entry⟩
The name of the entry to unlabel.

plotxyunselect

Unselects the given plots.

Syntax:

plotxyunselect ⟨plotname⟩

Operands:
⟨plotname⟩
Names of the plots to unselect, or all, or displayed.

plotxyupdate

Updates the selected plots.

Syntax:

plotxyupdate

plotxyzoom

Scales the selected plots.

Syntax:

plotxyzoom ⟨x⟩ ⟨y⟩

Operands:
⟨x⟩ ⟨y⟩
This is the amount of scaling to use in the horizontal direction. This is the amount of scaling to use in the vertical direction.

plotxyzoompan

Scales and offsets the given plot.

Syntax:

plotxyzoompan ⟨plot⟩ ⟨zoomx⟩ ⟨zoomy⟩ ⟨panx⟩ ⟨pany⟩

Operands:
⟨plot⟩ ⟨zoomx⟩ ⟨zoomy⟩ ⟨panx⟩ ⟨pany⟩
This is the name of the plot to apply the zoom and pan factors to. This is the amount of scaling to use in the horizontal direction. This is the amount of scaling to use in the vertical direction. This is the amount to offset in the horizontal direction as a percentage of the full range. This is the amount to offset in the vertical direction as a percentage of the full range.

posereceptor

Define attribute of pose receptor in the pose viewer.

Syntax:

posereceptor *display*=yes | no

Options:

display If this option is set to "false" then the pose file receptor will not be displayed in the 3D workspace.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

potential

Set various options associated with the definition of the potential energy to be used in a MacroModel job.

Syntax:

potential *cele*=⟨x⟩ *charges*=force_field | structure_file *chnd*=⟨x⟩ *cutoff*=normal | extended | user_defined | none *cvdw*=⟨x⟩ *dielectric*=⟨x⟩ *electrostatics*=field_field | constant | distance_dependant *field*=mm2* | mm3* | amber* | opls* | amber94 | mmff | mmffs | oplsaa | opls2003 *solvent*=none | water | chcl3 | octanol *substructure*=yes | no

Options:

cele This option determines what cutoff will be used for the electrostatic part of the energy calculation.

Valid values: reals
Default value: **12**
Minimum: 0.0
Maximum: 99999.0

charges This option determines where the charges to be used in the energy calculation will come from.

Valid values: force_field
structure_file
Default value: **force_field**

chnd This option determines what cutoff will be used for the hydrogen bond part of the energy calculation.

Valid values:	reals
Default value:	**4**
Minimum:	0.0
Maximum:	99999.0

cutoff This option determines what type of non-bonded cutoff will be used in the energy calculation.

Valid values:	normal extended user_defined none
Default value:	**normal**

cvdw This option determines what VDW cutoff will be used in the energy calculation.

Valid values:	reals
Default value:	**7**
Minimum:	0.0
Maximum:	99999.0

dielectric The dielectric constant to be used in the electrostatic part of the energy calculation.

Valid values:	reals
Default value:	**1**
Minimum:	0.9999999999

electrostatics
The electrostatic treatment to be used in the energy calculation.

Valid values:	field_field constant distance_dependant
Default value:	**constant**

field The force field to be used for the energy calculation.

Valid values:	mm2* mm3* amber* opls* amber94 mmff mmffs oplsaa opls2003
Default value:	**mmffs**

solvent The solvent model to be used for the energy calculation

Valid values: none
water
chcl3
octanol
Default value: **none**

substructure
[NOTE: This option is no longer used.]

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

pprep

This keyword is used to set various options associated with running protein preparation.

Syntax:

pprep *cavity*=default | liaison | no_neutralization *procedure*=both | prepare | refine *rmsd*=⟨ x ⟩

Options:

cavity Which preparation procedure to run. Both preparation and refinement by default.

Valid values: default
liaison
no_neutralization
Default value: **default**

procedure Which preparation procedure to run. Both preparation and refinement by default.

Valid values: both
prepare
refine
Default value: **both**

rmsd Minimum RMSD.

Valid values: reals
Default value: **0.3**
Minimum: 0.0

pprepreceptorligand

Defines a on-screen molecule to be treated as the ligand for a Pprep calculation

Syntax:

pprepreceptorligand ⟨ molecule_number ⟩

Operands:

⟨ molecule_number ⟩

The number of a molecule to be included as the ligand.

pprepwrite

Write the files required for protein structure preparation.

Syntax:

pprepwrite

prefer

Set the global preferences for handling markers (derived graphical objects). Specifically, specify under which conditions markers will be deleted.

Syntax:

prefer *autofit*=never | singleentry | always *beep*=yes | no *buildbackbonesubjobs*=⟨ n ⟩ *collapsed*=⟨ n ⟩ *commandinputshow*=yes | no *deletemarkers*=mismatch | missing *feedbackproperty*=⟨ text ⟩ *feedbackshow*=yes | no *initworkdir*=⟨ text ⟩ *keepphasejobfiles*=yes | no *lastphasedb*=⟨ text ⟩ *lastproject*=⟨ text ⟩ *macromodelsuffix*=⟨ text ⟩ *maestrosuffix*=⟨ text ⟩ *mol2suffix*=⟨ text ⟩ *openlastproject*=yes | no *pdbsuffix*=⟨ text ⟩ *projectsave*=small | medium *projectsuffix*=⟨ text ⟩ *projectsync*=auto | prompt | manual *rightclicktimerperiod*=⟨ n ⟩ *sdsuffix*=⟨ text ⟩ *sequenceviewerproximity*=⟨ x ⟩ *sequenceviewershow*=yes | no *showprojecttable*=yes | no *statusbarshow*=yes | no *tbcollapsed*=⟨ n ⟩ *tblocation*=⟨ n ⟩ *toolbarshow*=yes | no *topviewshow*=yes | no *workspacemenushow*=yes | no

Options:

autofit Specifies whether workspace entries fit to screen automatically or not. Valid values are "never", "singleentry", and "always". These cause Maestro never fit to screen, fit when only one entry in workspace and always fit to screen respectively.

Valid values: never
singleentry
always
Default value: **never**

beep Whether or not the system beeps for picking feedback.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

buildbackbonesubjobs
This is the number of templates to run at one time in Build Backbone.

Valid values: integers
Default value: **1**
Minimum: 1

collapsed Whether the status bar is collapsed or not.

Valid values: integers
Default value: **0**

commandinputshow
Whether the command input area is show or hide.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

deletemarkers

This option sets the condition for marker deletion. Valid values are "mismatch" or "missing".

Valid values: mismatch
missing
Default value: **mismatch**

feedbackproperty

This is the property displayed at the end of the feedback string in workspace.

Valid values: text strings
Default value: **Title**

feedbackshow

Whether the workspace feedback is activated.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

initworkdir

The Maestro input/output directory is used for starting jobs and for other file input and output. The i/o directory is always displayed in the title bar of the Main Application Window. The i/o dir can be set to a number of values which affect Maestro's behavior when reading and writing files and when running jobs. You can specify "currentdir" ("startdir" is a synonym) to set the i/o directory to use Maestro's current working directory (cwd), "project" to change to the project directory, "projectparent" to change to the parent directory which contains the project directory, or "projectjobs" to change to the "jobs" directory within the project directory. Otherwise, specify a directory path to be used. Maestro has a current working directory (cwd), similar to what a Unix shell has. When "currentdir" is in effect job files will be placed in the cwd. This is located at the very top of the window next to the window's border. The cwd is changed whenever a cd (changedirectory) command

Valid values: text strings
Default value: **currentdir**

keepphasejobfiles

If this is set to true, then Phase job files will be preserved after a job completes. Otherwise, the job files will be removed when each job completes.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

lastphasedb

This records the last Phase 3D Database which was opened.

Valid values: text strings
Default value:

lastproject This records the last project which was opened.

Valid values: text strings
Default value:

macromodelsuffix
The default suffix or extension to be used for exported MacroModel files. The suffix will automatically be appended to a file name if it has no suffix.

Valid values: text strings
Default value: **dat**

maestrosuffix
The default suffix or extension to be used for exported Maestro files. The suffix will automatically be appended to a file name if it has no suffix.

Valid values: text strings
Default value: **mae**

mol2suffix The default suffix or extension to be used for exported Mol2 files. The suffix will automatically be appended to a file name if it has no suffix.

Valid values: text strings
Default value: **mol2**

openlastproject
If set to true, then when Maestro starts it will re-open the last project that was open when Maestro was closed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

pdbsuffix The default suffix or extension to be used for exported PDB files. The suffix will automatically be appended to a file name if it has no suffix.

Valid values: text strings
Default value: **pdb**

projectsave
Specifies information to be saved to disk when project is closed. The "small" option saves only the compressed opening state. Because it takes longer to open and close projects when the current state must be expanded or deleted, this option is not recommended unless you are low on disk space. The "medium" option saves the compressed opening state and the expanded current project state. This option requires more disk space, but

provides faster and easier access and the safeguard of redundant project data. It also allows access to the project by 4.1 and 5.0 versions of Maestro that cannot use the compressed opening state.

Valid values: small
medium
Default value: **medium**

projectsuffix

The default suffix or extension to be used for project directories. When a project selector is used in Maestro to choose a project, the filter is automatically set to match this suffix. The suffix is automatically appended, if missing, to the returned project path when there is potential for creating a new project (projectnew, projectrename, or projectcopy). However, when a project command is issued without using a project selector (e.g. in a macro, script, or the command input area), the suffix is not automatically appended. Also, in cases where an existing project must be opened (projectopen, projectmerge) the suffix is not appended.

Valid values: text strings
Default value: **.prj**

projectsync

Specifies whether workspace changes to project entries are automatically saved or not. Valid values are "auto", "prompt", and "manual". These cause Maestro to save changes automatically, prompt to save changes, or save changes only when explicitly directed by the user.

Valid values: auto
prompt
manual
Default value: **auto**

rightclicktimerperiod

The time delay for workspace menu to appear in milliseconds.

Valid values: integers
Default value: **300**

sdsuffix

The default suffix or extension to be used for exported SD files. The suffix will automatically be appended to a file name if it has no suffix.

Valid values: text strings
Default value: **sdf**

sequenceviewerproximity

Cutoff distance for proximity coloring in the sequence viewer

Valid values: reals
Default value: **4**
Minimum: 0.0

sequenceviewershow
Whether the sequence viewer is show or hide.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

showprojecttable
If set to true, then when a project is opened, the project table will automatically be displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

statusbarshow
Whether the status bar is show or hide.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

tbcollapsed
Whether the tool bar is collapsed or not.

Valid values: integers
Default value: **0**

tblocation Whether the tool bar is located left or right.

Valid values: integers
Default value: **1**

toolbarshow
Whether the tool bar is show or hide.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

topviewshow
Whether the top view window is show or hide.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

workspacemenushow
Whether the workspace menu should be displayed on right click.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

previouspose

If there is more than one pose structure in the pose viewer, display the one before the first displayed pose. Upon reaching the beginning of the list, wraps around to the end.

Syntax:

previouspose

print

Print to a postscript printer or file. The print options must be set independently using the "printoptions" command. If an operand is given then the postscript is written to a file of that name, otherwise it is sent directly to the printer.

Syntax:

print [⟨file_name⟩]

Operands:

[⟨file_name⟩]

If an operand is given it is treated as the name of the file to which the postscript will be written. The full filename including any suffix should be specified.

printoptions

Set up options for the Print command, which sends the on-screen images to a postscript printer or file. If the destination is an encapsulated postscript file (eps), the command, orientation, scale_to_page, and title options are ignored

Syntax:

printoptions *color_mode*=color | black_text | black_text_and_lines *command*=⟨ text ⟩ *file_format*=ps | eps *orientation*=portrait | landscape *print_background*=yes | no *scale_to_page*=yes | no *smoothness*=⟨ n ⟩ *title*=⟨ text ⟩

Options:

color_mode

Specifies one of three modes: 1) full color (grey-scale on a black and white printer); 2) black text; and 3) black text and lines.

Valid values: color
black_text
black_text_and_lines
Default value: **color**

command This is the command used to print to a postscript printer.

Valid values: text strings
Default value: **lpr**

file_format

When printing to a file, specifies whether to generate postscript (ps) or encapsulated postscript (eps).

Valid values: ps
eps
Default value: **ps**

orientation

Specifies portrait or landscape orientation (ignored in eps files).

Valid values: portrait
landscape
Default value: **portrait**

print_background

Specifies whether or not to print the background color.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

scale_to_page

Specifies whether or not to scale the screen image to fit page (ignored in eps files).

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

smoothness

Determines how well smoothly shaded surfaces are renderered. The highest value (10) will result in the best image, however, the lowest value (1) will result in a shorter print file and faster printing.

Valid values: integers
Default value: **6**
Minimum: 1
Maximum: 10

title Optional title text, which is added to the bottom of the page (ignored in eps files).

Valid values: text strings
Default value:

projectclose

Close the current project and open a new scratch project.

Syntax:

projectclose

projectcopy

Make a copy of the current project.

Syntax:

projectcopy *entry*=yes | no *job*=yes | no *plot*=yes | no *run*=yes | no *snapshot*=yes | no *table*=yes | no *user*=yes | no ⟨to_dir_path⟩

Options:

entry Enable/disable copying of project entry files.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

job Enable/disable copying of project job files.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

plot Enable/disable copying of project plots. If this option is enabled, entries should also be copied.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

run Enable/disable copying of project run files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

snapshot Enable/disable copying of project snapshot.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

table Enable/disable copying of project table files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

user Enable/disable copying of project user files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ to_dir_path ⟩
The path (name and location) of the directory to be used to copy the current project. This can either be a new or existing project directory.

projectdelete

Delete the current project and open a new scratch project.

Syntax:

projectdelete

projectmerge

Merge data from another project into the current Maestro project.

Syntax:

projectmerge *entry*=yes | no *job*=yes | no *run*=yes | no
user=yes | no ⟨ from_dir_path ⟩

Options:

entry Enable/disable merging of project entry files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

job Enable/disable merging of project job files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

run Enable/disable merging of project run files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

user Enable/disable merging of project user files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨from_dir_path⟩
The path (name and location) of the project directory to be merged into the current project.

projectnew

Create a new project and open it in Maestro.

Syntax:

projectnew ⟨dir_path⟩

Operands:
⟨dir_path⟩
The path (name and location) of the directory to be created as the new project directory.

projectopen

Open an existing project into Maestro.

Syntax:

projectopen ⟨dir_path⟩

Operands:
⟨dir_path⟩
The path (name and location) of the project directory to be opened.

projectrename

Rename the current project directory and/or move it to a new location.

Syntax:

projectrename ⟨to_dir_path⟩

Operands:
⟨to_dir_path⟩
The new path (name and location) for the current project directory.

projectrevertopen

Revert state of current project to that it had when opened in Maestro.

Syntax:

projectrevertopen

projectrevertsnapshot

Revert state of current project to that previously stored in snapshot.

Syntax:

projectrevertsnapshot

projectsaveas

Save project and place user into that project

Syntax:

projectsaveas ⟨ to_dir_path ⟩

Operands:
⟨ to_dir_path ⟩
Name to which the project will be saved. Saves all data from the current project (table, plots, etc.)

projectstoresnapshot

Save copy of current project state for reversion.

Syntax:

projectstoresnapshot

projectsynchronize

Save changes in workspace to current project.

Syntax:

projectsynchronize

projecttablefind

Searches for a string in the project table and make that cell editable.

Syntax:

projecttablefind *direction*=up | down *matchcase*=yes | no *matchword*=yes | no ⟨find_string⟩

Options:

direction With this option user can specify the direction of search.

Valid values: up
down
Default value: **down**

matchcase With this option user can specify for case sensitive search.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

matchword
With this option user can specify for matching word as a whole.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨find_string⟩
Its the string for which the user want to search in the project table.

projectupdatecoordinates

Update the coordinates for the included entries in the project. This applies any current transformations in the Workspace to the original coordinates and places the result back into the project, overwriting the original coordinates.

Syntax:

projectupdatecoordinates

propertycalculate

This command calculates the given property for the for the entries that are specified with the ESL.

Syntax:

propertycalculate *propertyname*=⟨ text ⟩ *recalculate*=yes | no
⟨ ESL ⟩

Options:

propertyname
This option is the name of the property that need to be calculated. The valid values for this option are numatoms , nummolecules , molweight , spin , molcharge and secstruct for calculating the properties Number of atoms , Number of residues , Number of molecules , Molecular weight , Spin multiplicity , Molecular charge and Secondary Structure Content respectively. A new property will be added to the project this <propertyname> and the values will be assigned to those entries which are match the ESL.

Valid values: text strings
Default value: **numatoms**

recalculate The valid values are true and false. With this option the user can specify whether to re-calculate the property data, which has been calculated already. This option will not have any impact on the entries which does not have the property data. If this option value is true, then the property value will be calculated and the new value will be assigned. If the options value is false, then the already existing property values will not be calculated.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ ESL ⟩
A valid ESL specification. Calculates specified property for those entries which match the ESL description.

propertycalculatemolformula

This command calculates the molecular formula for the entries that are specified with the ESL.

Syntax:

propertycalculatemolformula *maxatoms*=⟨ n ⟩
recalculate=yes | no ⟨ ESL ⟩

Options:

maxatoms This option is used for calculating molecular formula for an entry which has only one molecule. If the entry contains more than the specified maxatom then the molecular formula will not be calculated for that entry.

Valid values: integers
Default value: **100**

recalculate The valid values are true and false. With this option the user can specify whether to re-calculate the property data, which has been calculated already. This option will not have any impact on the entries which does not have the property data. If this option value is true, then the property value will be calculated and the new value will be assigned. If the options value is false, then the already existing property values will not be calculated.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ ESL ⟩
A valid ESL specification. Calculates molecular formula for those entries which match the ESL description.

propertycalculatepickpka

This command calculates the pickPka atom for the entries that are specified with the ESL.

Syntax:

propertycalculatepickpka *atomname*=⟨ text ⟩
recalculate=yes | no ⟨ ESL ⟩

Options:

atomname This option is the name of the atom that is to be set as the pKa atom for the Jaguar calculation.

Valid values: text strings
Default value:

recalculate The valid values are true and false. With this option the user can specify whether to re-calculate the property data, which has been calculated already. This option will not have any impact on the entries which does not have the property data. If this option value is true, then the property value will be calculated and the new value will be assigned. If the options value is false, then the already existing property values will not be calculated.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ESL⟩
A valid ESL specification. Selects the pickPka atom for those entries which match the ESL description.

propertycalculatesubstructs

This command calculates the number of sub-structures for the entries that are specified with the ESL.

Syntax:

propertycalculatesubstructs *name*=⟨text⟩
recalculate=yes | no *substructure*=⟨text⟩ ⟨ESL⟩

Options:

name This option holds the name of substructure. A valid property name is the valid value for this option. A new property of the integer type will be created with this <name>, which holds the count of specified sub-structures in an entry.

Valid values: text strings
Default value:

recalculate The valid values are true and false. With this option the user can specify whether to re-calculate the property data, which has been calculated already. This option will not have any impact on the entries which does not have the property data. If this option value is true, then the property value will be calculated and the new value will be assigned. If the options value is false, then the already existing property values will not be calculated.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

substructure

This describes the sub-structure defintion. This value is used for matching substructures in each entry. The count of these sub-structures in each entry is saved as a property in the table.

Valid values: text strings
Default value:

Operands:
⟨ESL⟩
A valid ESL specification. Calculates the number of sub-structures for those entries which match the ESL description.

propertyclearvalue

This command clear values of the given properties

Syntax:

propertyclearvalue *allentries*=false | true ⟨propertynames⟩

Options:

allentries The property value to be cleared for selected entries or all the entries. This has two valid values, true for clearing all property values and false for clearing property values for the selected entries only.

Valid values: false
true
Default value: **false**

Operands:
⟨propertynames⟩
The names of properties for which values need to be cleared.

propertycreate

Create a new property for entries in the current project without assigning any entry property values.

Syntax:

propertycreate *author*=⟨ text ⟩ *displayprecision*=⟨ n ⟩ *type*=bool | int | double | string ⟨ property_name ⟩

Options:

author The m2io signature of the owner (e.g. person, group, or software module), or authority, that defines the property (meaning and range of values) and generally assigns its values. The author for user-defined properties should normally be user , as these properties can be freely edited in the project table. Be careful if setting author to m , mmod , i , j , qp , sd , or other names reserved for properties generated by existing programs.

Valid values: text strings
Default value: **user**

displayprecision
The Display precision for the real data type variable for display use only.

Valid values: integers
Default value: **6**
Minimum: 0
Maximum: 6

type The manner in which the property values are to be stored and represented, either bool (for Boolean), int (for integer), double (for double precision floating point, real numbers), or string (for text character strings).

Valid values: bool
int
double
string
Default value: **int**

Operands:

⟨ property_name ⟩

The user-assigned name of the property, which propertyrename can alter. The property name must be unique within the current project. If it is, a unique m2io data name will be generated by combining the type, author, and name, possibly modified (such as replacing spaces with underscores and adding a number at the end) to make it valid and unique.

propertydelete

This command deletes the given properties.

Syntax:

propertydelete ⟨propertynames⟩

Operands:
⟨propertynames⟩
The names of properties to delete.
Aliases:
deleteproperty (see [deleteproperty], page 74)

propertygeneratecontacts

This command applies a Contacts measurement as a property to a number of selected entries in a project. It is valid at least Contactset1 has been set. If onle Contactset1 is defined, then Contactset2 = Contactset1. Normally these two sets are required having been set before this command, otherwise no property will be generated.

Syntax:

propertygeneratecontacts

propertygeneratehbond

This command applies an HBond measurement as a property to a number of selected entries in a project. It is valid at least HBondset1 has been set. If onle HBondset1 is defined, then HBondset2 = HBondset1. Normally these two sets are required having been set before this command, otherwise no property will be generated.

Syntax:

propertygeneratehbond

propertymeasurementsetting

This command sets whether a measurement is applied as a property to a number of selected entries in a project.

Syntax:

propertymeasurementsetting *applytoselectedentries*=yes | no

Options:

applytoselectedentries
: This option determines whether to apply a measurement as a property to a number of selected entries in a project.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

propertymove

Moves the property in the first column to the second column.

Syntax:

propertymove ⟨from⟩ ⟨to⟩ ⟨table⟩

Operands:

⟨from⟩ ⟨to⟩ ⟨table⟩

The source column number. The destination column number. The name of the table to use as the source for the column numbers. If the table operand is missing, the current or default table (1) will be used.

propertyprecision

This command sets the precision of the property.

Syntax:

propertyprecision ⟨property_name⟩ ⟨precision⟩

Operands:
⟨property_name⟩ ⟨precision⟩
The name of the property. Precision for that property.

propertyrename

This command renames the given property to the given name.

Syntax:

propertyrename ⟨from⟩ ⟨to⟩

Operands:
⟨from⟩ ⟨to⟩
The name of the property to rename. The name to rename the property to.
Aliases:
renameproperty (see [renameproperty], page 362)

propertyshowall

This command creates a property subset consisting of all the properties in the project.

Syntax:

propertyshowall

propertysuperimposesetting

This command sets whether a superimposition is applied as a property to a number of included entries in a project.

Syntax:

propertysuperimposesetting *applytoincludedentries*=yes | no

Options:

applytoincludedentries
This option determines whether to apply a superimposition as a property to a number of included entries in a project.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

pspalignaddanchor

Adds an anchor at the given position for Edit Alignment.

Syntax:

pspalignaddanchor ⟨anchor position⟩

Operands:
⟨anchor position⟩
The position at which to add an anchor.

pspaligndeleteanchor

Deletes the anchor at the given position for Edit Alignment.

Syntax:

pspaligndeleteanchor ⟨anchor position⟩

Operands:
⟨anchor position⟩
The position at which to delete an anchor.

pspaligninsertgaps

Inserts gaps into the sequence at the given position.

Syntax:

pspaligninsertgaps *number*=⟨ n ⟩ *position*=⟨ n ⟩ ⟨ sequence name ⟩

Options:

number This is the number of gaps to insert.

Valid values: integers
Default value: **1**
Minimum: 1

position This is the position at which to insert a gap.

Valid values: integers
Default value: **1**
Minimum: 1

Operands:
⟨ sequence name ⟩
The name of the sequence to insert gaps into.

pspalignlockgaps

Locks gaps for all alignments in the Edit Alignment step.

Syntax:

pspalignlockgaps

pspalignmoveleft

Moves the alignment left freely starting at the given position. This will shift gaps from the left of the given position to the right of the given position.

Syntax:

pspalignmoveleft *number*=⟨n⟩ *position*=⟨n⟩ ⟨sequence name⟩

Options:

number This is the number of spaces to move left.

Valid values: integers
Default value: **1**
Minimum: 1

position This is the position at which to move left.

Valid values: integers
Default value: **1**
Minimum: 1

Operands:
⟨sequence name⟩
The name of the sequence to move left.

pspalignmoveleftblock

Moves the alignment left as a block starting at the given position. This will close up gaps to the left of the given position while preserving them to the right.

Syntax:

pspalignmoveleftblock *number*=⟨n⟩ *position*=⟨n⟩
⟨sequence name⟩

Options:

number This is the number of spaces to move left.

Valid values: integers
Default value: **1**
Minimum: 1

position This is the position at which to move left.

Valid values: integers
Default value: **1**
Minimum: 1

Operands:

⟨sequence name⟩

The name of the sequence to move left.

pspalignmoveright

Moves the alignment right freely starting at the given position. This will shift gaps from the right of the given position to the left of the given position. If necessary, this will open up gaps to the left of the given position.

Syntax:

pspalignmoveright *number*=⟨n⟩ *position*=⟨n⟩ ⟨sequence name⟩

Options:

number This is the number of spaces to move right.

Valid values:	integers
Default value:	**1**
Minimum:	1

position This is the position at which to move right.

Valid values:	integers
Default value:	**1**
Minimum:	1

Operands:

⟨sequence name⟩

The name of the sequence to move right.

pspalignstructures

Runs 'structalign' to align the selected template structures in the Find Homologs step.

Syntax:

pspalignstructures

pspalignunlockgaps

Unloocks gaps for all alignments in the Edit Alignment step.

Syntax:

pspalignunlockgaps

pspbstogglehetatom

Toggles the given hetatom between included and excluded.

Syntax:

pspbstogglehetatom ⟨hetatom name⟩

Operands:
⟨hetatom name⟩
The name of the hetatom to toggle.

pspbuildbackbone

Runs the build backbone backend.

Syntax:

pspbuildbackbone

pspbuildstructure

Runs the build structure backend.

Syntax:

pspbuildstructure

pspexcludetable1row

Excludes a composite structure from the table shown in the Build Backbone step. Excludes a structure, for the specified row in the input table shown in the current Structure Prediction step, from the Sequence Viewer (and Workspace). This applies to the Refine Backbone step.

Syntax:

pspexcludetable1row ⟨ row_number ⟩

Operands:

⟨ row_number ⟩

The row number in the table which is to be excluded.

pspexcludetablerow

Excludes a composite structure from the table shown in the Build Backbone step. Excludes a structure, for the specified row in the table shown in the current Structure Prediction step, from the Sequence Viewer (and Workspace). This applies to the Fold Recognition, Build Backbone, Refine Backbone, and Refine Structure steps (which have structure or template tables, and the ability to specify rows in Workspace independently from selected rows).

Syntax:

pspexcludetablerow ⟨ row_number ⟩

Operands:

⟨ row_number ⟩

The row number in the table which is to be excluded.

pspexportalignment

exports alignments selected into a file.

Syntax:

pspexportalignment ⟨filename⟩

Operands:
⟨filename⟩
The name of the file to which the alignments are to be written.

pspfindfamily

Finds the family for the query sequence currently active in the sequence viewer.

Syntax:

pspfindfamily

pspfindhomologs

Runs Blast to find homologs for the query sequence currently active in the sequence viewer.

Syntax:

pspfindhomologs

pspfoldrecognitionoptions

Sets some options associated with the Fold Recognition step.

Syntax:

pspfoldrecognitionoptions *max_results*=⟨ n ⟩

Options:

max_results
Specifies how many results will be shown in the fold recognition results table.

Valid values: integers
Default value: **100**

pspfoldrecognitionsearch

Runs the fold recognition search on the current query sequence.

Syntax:

pspfoldrecognitionsearch

pspimportalignment

imports alignments in a file and applies to the templates. since there can be more than one alignment stored in a file, the command gives messages on which templates have been altered.

Syntax:

pspimportalignment ⟨ filename ⟩

Operands:
⟨ filename ⟩
The name of the file from which the alignments are to be read.

pspimporthomolog

Imports the sequences from a file as homologs to the current query sequence.

Syntax:

pspimporthomolog ⟨filename⟩

Operands:

⟨filename⟩

The name of the file from which the sequences are to be read.

pspimportssp

Adds the secondary structure prediction from a file to those for the current query sequence.

Syntax:

pspimportssp ⟨filename⟩

Operands:

⟨filename⟩

The name of the file from which the SSP is to be read.

pspincludetable1row

Includes a structure from the input table shown in the current Structure Prediction step into the Sequence Viewer (and Workspace), independent of selection. This applies to the Refine Backbone step.

Syntax:

pspincludetable1row ⟨row_number⟩

Operands:

⟨row_number⟩

The row number in the table which is to be included.

pspincludetablerow

Includes a structure from the table shown in the current Structure Prediction step into the Sequence Viewer (and Workspace). This applies to the Fold Recognition, Build Backbone, Refine Backbone, and Refine Structure steps (which have structure or template tables, and the ability to specify rows in Workspace independently from selected rows).

Syntax:

pspincludetablerow ⟨ row_number ⟩

Operands:
⟨ row_number ⟩
The row number in the table which is to be included.

pspminimizationresidues

Defines a set of atoms for minimization refinement.

Syntax:

pspminimizationresidues ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
The ASL expression which defines the atoms that will be used to define the residues for minimization refinement.

pspoptimizealignment

optimizes alignment of a template sequence.

Syntax:

pspoptimizealignment ⟨ rowindex ⟩

Operands:
⟨ rowindex ⟩

The row index of the sequence which is aligned

psprefinebackbone

Runs the refine backbone backend.

Syntax:

psprefinebackbone

psprefinestructure

Runs the structure refinement program with the currently set refinement options on the currently selected structure.

Syntax:

psprefinestructure

psprsdefaulthelixloops

Rebuilds the Helices & Loops table for structure refinement with the default loop and helix refinement options for the currently selected structure.

Syntax:

psprsdefaulthelixloops

psprshelixlooptogglerefine

Toggle refinement of loop or helix.

Syntax:

psprshelixlooptogglerefine ⟨row_index⟩

Operands:

⟨row_index⟩

The row index for the loop or helix in the Helices & Loops table that is to be toggled for refinement.

psprshelixoptions

Specifies the refinement options for the currently selected helix in the Helices & Loops table.

Syntax:

psprshelixoptions *roll_range*=⟨x⟩ *roll_resolution*=⟨x⟩ *set_roll*=yes | no *set_shift*=yes | no *set_tilt*=yes | no *set_trans*=yes | no *shift_range*=⟨x⟩ *shift_resolution*=⟨x⟩ *tilt_range*=⟨x⟩ *tilt_resolution*=⟨x⟩ *trans_range*=⟨x⟩ *trans_resolution*=⟨x⟩

Options:

roll_range Range, in degrees, for helix roll.

Valid values: reals
Default value: **20**

roll_resolution
Resolution, in degrees, for helix roll.

Valid values: reals
Default value: **20**

set_roll Set the helix roll range and resolution.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

set_shift Set the helix shift range and resolution.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

set_tilt Set the helix tilt range and resolution.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

set_trans Set the helix translation range and resolution.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

shift_range
Range, in angstroms, for helix shift.

Valid values: reals
Default value: **2**

shift_resolution
Resolution, in angstroms, for helix shift.

Valid values: reals
Default value: **2**

tilt_range Range, in degrees, for helix tilt.

Valid values: reals
Default value: **5**

tilt_resolution
Resolution, in degrees, for helix tilt.

Valid values: reals
Default value: **5**

trans_range
Range, in angstroms, for helix translation.

Valid values: reals
Default value: **2**

trans_resolution
Resolution, in angstroms, for helix translation.

Valid values: reals
Default value: **2**

psprsloopoptions

Specifies the refinement options for the currently selected loop in the Helices & Loops table.

Syntax:

psprsloopoptions *ca_max*=⟨ x ⟩ *overlap_min*=⟨ x ⟩
restrict_ca=yes | no *sphere_size*=⟨ x ⟩ *use_sphere*=yes | no

Options:

ca_max Maximum CA atom movement, in angstroms, from initial.

Valid values: reals
Default value: **3**

overlap_min
Minimum overlap, in angstroms, from initial.

Valid values: reals
Default value: **0.7**
Minimum: 0.1
Maximum: 1.0

restrict_ca Restrict loop CA atom movement.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

sphere_size
Distance from loop for sidechain inclusion, in angstroms.

Valid values: reals
Default value: **7.5**

use_sphere Include nearby sidechains in loop refinement.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

psprunalign

Runs secondary structure prediction (if necessary) and then aligns the query to the active templates in Edit Alignment.

Syntax:

psprunalign

pspruncreate

Creates the run with the given name.

Syntax:

pspruncreate ⟨run name⟩

Operands:
⟨run name⟩
The name of the new run to create.

psprundelete

Deletes the current run from the project.

Syntax:

psprundelete

psprunopen

Opens the run with the given name.

Syntax:

psprunopen ⟨run name⟩

Operands:
⟨run name⟩
The name of the run to open.

psprunrename

Changes the current run's name to the given name.

Syntax:

psprunrename ⟨run name⟩

Operands:
⟨run name⟩
The name to change the current run's name to.

psprunsaveas

Saves a copy of the current run under the given name.

Syntax:

psprunsaveas ⟨run name⟩

Operands:
⟨run name⟩
The name of the run to save as.

pspsecstructprediction

Runs the available secondary structure prediction programs on the current query sequence.

Syntax:

pspsecstructprediction

pspselectextendtable1row

Extend the selection from the selected table row to joing up with an existing selection. This applies to the Refine Backbone step

Syntax:

pspselectextendtable1row ⟨row_number⟩

Operands:

⟨row_number⟩

The row number in the table from which the selection is to begin.

pspselectextendtablerow

Extend the selection from the selected table row to joing up with an existing selection. This applies to the Find Homologs, Fold Recognition, Build Backbone, Refine Backbone, Edit Alignment and Refine Structure steps.

Syntax:

pspselectextendtablerow ⟨row_number⟩

Operands:

⟨row_number⟩

The row number in the table from which the selection is to begin.

pspselecthelixlooprow

Select structure for refinement.

Syntax:

pspselecthelixlooprow ⟨row_index⟩

Operands:

⟨row_index⟩

The row index for the loop or helix in the Helices & Loops table that is to be selected for refinement option editing.

pspselectonlytable1row

Select a row from the input structure or template table shown in the current Structure Prediction step, deselecting all other rows. This applies to the Refine Backbone step.

Syntax:

pspselectonlytable1row ⟨row_number⟩

Operands:
⟨row_number⟩
The row number in the table which is to be selected.

pspselectonlytablerow

Select a row from the structure or template table shown in the current Structure Prediction step, deselecting all other rows. This applies to the Find Homologs, Fold Recognition, Build Backbone, Refine Backbone, Edit Alignment and Refine Structure steps.

Syntax:

pspselectonlytablerow ⟨row_number⟩

Operands:
⟨row_number⟩
The row number in the table which is to be selected.

pspselectrscontext

Specify the context for structure refinement. This command may be executed instead of showing the psp or refinement panels. Be cautious about using this commands while either of these panels is shown, as setting the wrong context will interfere with the execution of the refinement commands issued from the panel.

Syntax:

pspselectrscontext pspstep|standalone

Operands:
pspstep|standalone
The context in which refine structure commands are executed.

pspselectrsrefinement

Selects the current psp structure refinement type.

Syntax:

pspselectrsrefinement ⟨refine_type⟩

Operands:
⟨refine_type⟩
The type of structure refinement to be performed. Value should be sidechains , helixloops , or minimization .

pspselecttable1row

Selects a row from the input structure or template table shown in the current Structure Prediction step. This applies to the Refine Backbone step.

Syntax:

pspselecttable1row ⟨row_number⟩

Operands:
⟨row_number⟩
The row number in the table which is to be selected.

pspselecttablerow

Selects a row from the structure or template table shown in the current Structure Prediction step. This applies to the Find Homologs, Fold Recognition,

Build Backbone, Refine Backbone, Edit Alignment and Refine Structure steps.

Syntax:

pspselecttablerow ⟨row_number⟩

Operands:
⟨row_number⟩
The row number in the table which is to be selected.

pspsequenceaddfile

Adds the sequences from the given file to the Select Sequence step in PSP.

Syntax:

pspsequenceaddfile ⟨file⟩

Operands:
⟨file⟩
The file name of the sequence file to add.

pspsequenceaddworkspace

Adds the sequences from the Workspace to the Select Sequence step in PSP.

Syntax:

pspsequenceaddworkspace

pspsequencecrop

Crops the given sequence to the given residue index, towards the closer end of the sequence.

Syntax:

pspsequencecrop *res*=⟨n⟩ ⟨sequence name⟩ ⟨residue index⟩

Options:

res This is the position to crop to

Valid values:	integers
Default value:	**1**
Minimum:	1

Operands:
⟨sequence name⟩ ⟨residue index⟩
The name of the sequence to crop.

pspsequenceselect

Selects the given sequence number as input for the next step in PSP.

Syntax:

pspsequenceselect ⟨sequence number⟩

Operands:
⟨sequence number⟩
The index of the sequence to select.

pspsequenceviewerexport

Exports all of the visible sequences in the sequence viewer to the given file.

Syntax:

pspsequenceviewerexport ⟨file name⟩

Operands:
⟨file name⟩
The name of the file to export the sequences to.

pspsethelixloopresidues

Specify residues to define loop or helix feature for structure refinement.

Syntax:

pspsethelixloopresidues ⟨row_index⟩ ⟨R1⟩ ⟨R2⟩ ⟨R3⟩ ⟨R4⟩

Operands:
⟨row_index⟩ ⟨R1⟩ ⟨R2⟩ ⟨R3⟩ ⟨R4⟩
The row index for the loop or helix in the Helices & Loops table, followed by the strings for the four residue column values, as they are to appear in the table. For a loop, only two end residues (<R1> <R2>) are required, so the <R3> and <R4> values can be missing or left blank. For a helix, <R1> and <R4> specify the outer residues for the loops at the two ends of the helix, while <R2> and <R3> specify the ends of the helix itself. The format for the residue strings is the residue number followed by the insertion code, if any. For example, 12C represent residue number 12, insertion code C.

pspsidechainresidues

Defines a set of atoms for the side chain refinement.

Syntax:

pspsidechainresidues ⟨ASL⟩

Operands:
⟨ASL⟩
The ASL expression which defines the atoms that will be used to define the residues for side chain refinement.

pspsortbbtable

Resort the Build Backbone table based on the data in the specified column

Syntax:

pspsortbbtable ⟨column_name⟩

Operands:
⟨column_name⟩
The name of the column to be sorted.

pspsortfindhomologstable

Resort the find homologs table based on the data in the specified column

Syntax:

pspsortfindhomologstable ⟨column_name⟩

Operands:
⟨column_name⟩
The name of the column to be sorted.

pspsortfoldtable

Resort the fold recognition table based on the data in the specified column

Syntax:

pspsortfoldtable ⟨column_name⟩

Operands:
⟨column_name⟩
The name of the column to be sorted.

pspsortrbtable

Resort the Refine Backbone output structure table based on the data in the specified column

Syntax:

pspsortrbtable ⟨column_name⟩

Operands:
⟨column_name⟩
The name of the column to be sorted.

pspsortrbtable1

Resort the Refine Backbone composite structure table based on the data in the specified column

Syntax:

pspsortrbtable1 ⟨column_name⟩

Operands:
⟨column_name⟩
The name of the column to be sorted.

pspsortrstable

Resort the Refine Structure table based on the data in the specified column

Syntax:

pspsortrstable ⟨column_name⟩

Operands:
⟨column_name⟩
The name of the column to be sorted.

pspsspdelete

Deletes the given secondary structure prediction.

Syntax:

pspsspdelete ⟨SSP name⟩

Operands:
⟨SSP name⟩
The name of the secondary structure prediction to delete.

pspsspexport

Exports the given secondary structure to the named file.

Syntax:

pspsspexport *ssp*=⟨text⟩ ⟨file name⟩

Options:

ssp This option is the name of the secondary structure prediction to export.
Valid values: text strings
Default value:

Operands:
⟨file name⟩
The name of the file to export to.

pspssprevert

Reverts a modified secondary structure prediction back to its original form.

Syntax:

pspssprevert ⟨SSP name⟩

Operands:
⟨SSP name⟩
The name of the secondary structure prediction to revert.

pspsspset

Sets the given range of positions in the secondary structure position to the given code, which must be H , E , or - .

Syntax:

pspsspset *code*=⟨text⟩ *from*=⟨n⟩ *to*=⟨n⟩ ⟨SSP name⟩

Options:

code This is the code for setting the secondary structure position to. It must be one of H , E , or - .

Valid values: text strings
Default value:

from This is the starting position for setting codes.

Valid values: integers
Default value: **1**
Minimum: 1

to This is the ending position for setting codes.

Valid values: integers
Default value: **1**
Minimum: 1

Operands:
⟨SSP name⟩
The name of the step to set data for.

pspstepforward

Moves forward to the next named step. Deletes any steps after the current step and copies data forward to create the specified step.

Syntax:

pspstepforward ⟨step name⟩

Operands:
⟨step name⟩
The name of the step to switch to.

pspstepgoto

Moves to an existing step in the current run.

Syntax:

pspstepgoto ⟨step name⟩

Operands:
⟨step name⟩
The name of the step to switch to.

pspstructureaddentry

Create project entry from selected psp structure. This operation is only permitted for the Refine Backbone and Refine Structure steps.

Syntax:

pspstructureaddentry ⟨row_index⟩ ⟨entry_name⟩

Operands:
⟨row_index⟩ ⟨entry_name⟩
The optional operands are the row index of the psp refined structure and the entry base name from which a unique name will be generated for the new project entry. The row is the full, not the visible, row index into the table of structures for the current Structure Prediction step. If the operands are missing, the selected row(s) from the table will be used to create project entries, using the row structure name to generate the entry name.

psptemplatesetregion

Sets the region for the given template.

Syntax:

psptemplatesetregion *end*=⟨ n ⟩ *start*=⟨ n ⟩ ⟨ template name ⟩

Options:

end This is the position at which the region ends.

Valid values: integers
Default value: **1**
Minimum: 1

start This is the position at which to start the region.

Valid values: integers
Default value: **1**
Minimum: 1

Operands:
⟨ template name ⟩
The name of the template to set a region for.

psptmplsecstructprediction

Runs the available secondary structure prediction programs on the template sequence identified by row number

Syntax:

psptmplsecstructprediction ⟨ row_number ⟩

Operands:
⟨ row_number ⟩
The row number in the table which is to be selected.

pspunselecttable1row

Unselects a row from the input table shown in the current Structure Prediction step. This applies to the Refine Backbone step.

Syntax:

pspunselecttable1row ⟨row_number⟩

Operands:

⟨row_number⟩

The row number in the table which is to be unselected.

pspunselecttablerow

Unselects a row from the table shown in the current Structure Prediction step. This applies to the Find Homologs, Fold Recognition, Build Backbone, Refine Backbone, and Edit Alignment steps (which support multiple row selection).

Syntax:

pspunselecttablerow ⟨row_number⟩

Operands:

⟨row_number⟩

The row number in the table which is to be unselected.

pspupdatescores

Updates Scores of all the alignments

Syntax:

pspupdatescores

pyeval

This is a standard alias for **pythoneval** (see [pythoneval], page 353).

pyimp

This is a standard alias for **pythonimport** (see [pythonimport], page 353).

pyrun

This is a standard alias for **pythonrun** (see [pythonrun], page 354).

pythoneval

Evaluate the python expression using the built-in Python interpreter. KEY_OPERAND_SYNOPOSIS: <python expression>

Syntax:

pythoneval

Aliases:

pyeval (see [pyeval], page 352)

pythonimport

Imports the specified python module. If the module has already been imported then it will be reloaded. Under normal usage this command is not needed as pythonrun automatically imports the module specified as its operand. However during development of modules it is useful to be able to reload them using pythonimport KEY_OPERAND_SYNOPOSIS: <module>

Syntax:

pythonimport

Aliases:

pyimp (see [pyimp], page 353)

pythonrun

Runs the function in <function name> in module <modulename> KEY_OPERAND_SYNOPOSIS: <function name> [function parameters]

Syntax:

pythonrun
Aliases:
pyrun (see [pyrun], page 353)

qikprop

This keyword is used to set various options associated with running QikProp jobs.

Syntax:

qikprop *fastmode*=yes | no *structure_source*=selected_entries | workspace | file

Options:

fastmode The mode in which QP job runs. (fast/slow)

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

structure_source
Whether to use the selected entries in the current project or what is in the workspace as input for the job.

Valid values: selected_entries
workspace
file
Default value: **selected_entries**

qsarmarkerdump

Print out the current option values of the QSAR marker command.

Syntax:

qsarmarkerdump

qsarmarkersettings

Set graphical data of Phase QSAR markers.

Syntax:

qsarmarkersettings *ambient*=⟨ x ⟩ *diffuse*=⟨ x ⟩ *effects*=allclass | selectedclass *emission*=⟨ x ⟩ *negativecoefficient*=⟨ x ⟩ *numberpls*=⟨ n ⟩ *positivecoefficient*=⟨ x ⟩ *roundingeffect*=⟨ x ⟩ *selectedatomclass*=⟨ text ⟩ *shininess*=⟨ x ⟩ *specular*=⟨ x ⟩ *step*=⟨ n ⟩ *transparency*=⟨ x ⟩ *volumeoccupied*=workspacelig | qsarmodel

Options:

ambient Set material property - ambient, to its red, green, and blue components, for front face.

Valid values:	reals
Default value:	**0.4**
Minimum:	0.0
Maximum:	1.0

diffuse Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values:	reals
Default value:	**0.4**
Minimum:	0.0
Maximum:	1.0

effects Set the QSAR visualization option of viewing effects.

Valid values:	allclass selectedclass
Default value:	**allclass**

emission Set material property - emission, to its red, green, and blue components, for front face.

Valid values:	reals
Default value:	**0.1**
Minimum:	0.0
Maximum:	1.0

negativecoefficient
Set the QSAR visualization option of negative coefficient threshold.

Valid values: reals
Default value: **-0.044**
Maximum: 0.0

numberpls Set the QSAR visualization option of number of PLS factors.

Valid values: integers
Default value: **1**
Minimum: 1

positivecoefficient
Set the QSAR visualization option of positiv coefficient threshold.

Valid values: reals
Default value: **0.044**
Minimum: 0.0

roundingeffect
This determines the rounding effect of edges of a cell, for front face.

Valid values: reals
Default value: **5**
Minimum: 0.0

selectedatomclass
Set the QSAR visualization option of selected atom class.

Valid values: text strings
Default value: **D**

shininess Set material property - shininess, for front face.

Valid values: reals
Default value: **80**
Minimum: 0.0
Maximum: 128.0

specular Set material property - specular, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

step The step domain tolerance of cells.

Valid values: integers
Default value: **3**
Minimum: 1

transparency
The transparency of QSAR markers.

Valid values:	reals
Default value:	**50**
Minimum:	0.0
Maximum:	100.0

volumeoccupied
Set the QSAR visualization option of viewing volume occupied.

Valid values:	workspacelig qsarmodel
Default value:	**workspacelig**

qsiteion

Defines an ion as part of the QM region for an Impact QSite simulation

Syntax:

qsiteion *basis*=⟨ text ⟩ ⟨ atom_num ⟩

Options:

basis The basis set for the ligand to be included in the QM region of an Impact QSite simulation.

Valid values:	text strings
Default value:	**6-31G***

Operands:
⟨ atom_num ⟩
The atom number of an ion to be included in the QM region in a Impact QSite simulation.

qsiteresidue

Defines a residue as part of the QM region for an Impact QSite simulation

Syntax:

qsiteresidue *basis*=⟨ text ⟩ *cuttype*=⟨ n ⟩
⟨ chain ⟩:⟨ molnum ⟩:⟨ resnum ⟩:⟨ inscode ⟩

Options:

basis The basis set for the residue to be included in the QM region of an Impact QSite simulation.
Valid values: text strings
Default value: **6-31G***

cuttype The cut type for the residue to be included in the QM region of an Impact QSite simulation.
Valid values: integers
Default value: **1**
Minimum: 0
Maximum: 5

Operands:
⟨ chain ⟩:⟨ molnum ⟩:⟨ resnum ⟩:⟨ inscode ⟩
The number of a residue to be included in the QM region in a Impact QSite simulation.

qsiteset

Settings associated with QSite simulations in Impact.

Syntax:

qsiteset *charge*=⟨ n ⟩ *maxiter*=⟨ n ⟩ *method*=dft | hf | lmp2
multiplicity=⟨ n ⟩ *numberprocessors*=⟨ n ⟩
optimization=singlepoint | minimization | transitionstate
optimize=yes | no *options*=⟨ text ⟩ *pathfraction*=⟨ x ⟩
product_entry=⟨ text ⟩ *reactant_entry*=⟨ text ⟩
tsguess_entry=⟨ text ⟩ *tsmethod*=standard | lst | qst

Options:

charge The total charge on the QM part in an Impact QSite simulation.
Valid values: integers
Default value: **0**

maxiter The maximun number of iterations for the QM optimization.
Valid values: integers
Default value: **100**

method
The QM Method used in a QSite job in Maestro
Valid values: dft
hf
lmp2
Default value: **dft**

multiplicity
The multiplicity of the QM part in an Impact QSite simulation.
Valid values: integers
Default value: **1**

numberprocessors
The number of processors to be used for the Jaguar Part of the calculation.
Valid values: integers
Default value: **1**

optimization
What type of calculation is to be performed in the QM part.
Valid values: singlepoint
minimization
transitionstate
Default value: **minimization**

optimize
[NOTE: This option is no longer used.] An option which determines if the QM part will be geometry optimized during a QSite job.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

options
Any additional Jaguar options which can be used during the Impact QSite job.
Valid values: text strings
Default value: **iacc=1 vshift=1.0 maxit=100**

pathfraction
The fraction of the path between the reactant and product which the TS is along.
Valid values: reals
Default value: **0.5**
Minimum: -0.000000
Maximum: 1.0000001

product_entry
The name of the entry which represents the product in a transition state calculation.
Valid values: text strings
Default value:

reactant_entry
The name of the entry which represents the entry in a transition state calculation.

Valid values: text strings
Default value:

tsguess_entry
The name of the entry which represents the transtion state guess in a transition state calculation.

Valid values: text strings
Default value:

tsmethod For a transition state calculation, how that is to be performed.

Valid values: standard
lst
qst
Default value: **standard**

quit

Quit the program. To quit issue just the quit command without any options.

Syntax:

quit *confirm*=yes | no

Options:

confirm If this option has been set to "false" then the program will exit without prompting the user in any way. So, for example, the user is not prompted to save any changed macros nor will the Quit panel be displayed. Note that invoking quit with any options only sets the option it does not also try to quit Maestro.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

read

This is a standard alias for **fileread** (see [fileread], page 116).

readposefile

Read the receptor and ligand poses from pose file in the pose viewer.

Syntax:

readposefile ⟨file_name⟩

Operands:
⟨file_name⟩
The name of the file from which the structures will be read. If no name is specified, the current pose list will be cleared.

readpotential

Read potential settings from a command file.

Syntax:

readpotential ⟨file_name⟩

Operands:
⟨file_name⟩
The name of the file from which the potential settings will be read. If no name is specified, the default settings will be used.

refinestart

Start a Refine input file with the current settings.

Syntax:

refinestart

refinewrite

Write a Refine input file with the current settings.

Syntax:

refinewrite

rename

Rename a named object. The object type is the same as the command which is used to create that type of object.
For example to rename a set named "alpha" use: rename set alpha beta .

Syntax:

rename ⟨object_type⟩ ⟨current_object_name⟩ ⟨new_object_name⟩

Operands:

⟨object_type⟩ ⟨current_object_name⟩ ⟨new_object_name⟩

The first operand is the name of the existing object.

renameproperty

This is a standard alias for **propertyrename** (see [propertyrename], page 323).

repall

Set global representation properties

Syntax:

repall *ballhresolution*=⟨ n ⟩ *balllresolution*=⟨ n ⟩ *ballsize*=⟨ x ⟩
border=auto | on | true | yes | off | false | no
borderscale=⟨ x ⟩ *bstyle*=split | blend *cpkhresolution*=⟨ n ⟩
cpklresolution=⟨ n ⟩ *cpksize*=⟨ x ⟩ *resolution*=high | low
rstyle=multiple | thick *smooth*=yes | no *stickradius*=⟨ x ⟩
tuberadius=⟨ x ⟩ *wirethickness*=⟨ n ⟩

Options:

ballhresolution
Set Ball high resolution

Valid values: integers
Default value: **16**
Minimum: 4
Maximum: 53

balllresolution
Set Ball resolution

Valid values: integers
Default value: **4**
Minimum: 4
Maximum: 53

ballsize Percentage to draw balls out at

Valid values: reals
Default value: **25**
Minimum: 4.0
Maximum: 200.0

border Set drawing of bond order to be on, off, or automatically determined by viewing scale.

Valid values: auto
on
true
yes
off
false
no
Default value: **auto**

borderscale
The drawing scale at which bond orders will be displayed

Valid values: reals
Default value: **30**

bstyle Set the bond style

Valid values: split
blend
Default value: **split**

cpkhresolution
Set CPK high resolution

Valid values: integers
Default value: **20**
Minimum: 8
Maximum: 57

cpklresolution
Set CPK low resolution

Valid values: integers
Default value: **8**
Minimum: 8
Maximum: 57

cpksize Percentage to draw CPK spheres out at

Valid values: reals
Default value: **85**
Minimum: 4.0
Maximum: 200.0

resolution Set the overall resolution for drawing molecules. low or high.

Valid values: high
low
Default value: **high**

rstyle Set the bond render style

Valid values: multiple
thick
Default value: **multiple**

smooth Enable/disable line and polygon antialiasing

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

stickradius
Radius of of sticks in Angstroms

Valid values: reals
Default value: **0.15**
Minimum: 0.01
Maximum: 1.0

tuberadius Radius of of tubes in Angstroms

Valid values: reals

Default value: **0.25**
Minimum: 0.01
Maximum: 1.0

wirethickness
Thickness of wireframe lines

Valid values: integers
Default value: **2**
Minimum: 1
Maximum: 20

repatom

Change the representation used to display a group of atoms in the main structure window.

Syntax:

repatom *rep*=none | circle | cpk | ballnstick ⟨ASL⟩

Options:

rep Type of atom representation

Valid values: none
circle
cpk
ballnstick
Default value: **none**

Operands:

⟨ASL⟩

A string in the atom specification language. All atoms which match this specification will have their representation changed.

repatombonds

Set representation of all of atoms' bonds. Uses bond rep specified in the repbond command.

Syntax:

repatombonds ⟨ASL⟩

Operands:
⟨ASL⟩
A string in the atom specification language. All atoms which match this string will have the representation of all bonds to them changed.

repbond

Change the representation used to draw an on-screen bond.

Syntax:

repbond *rep*=none | wire | tube ⟨atom1⟩ ⟨atom2⟩

Options:

rep	Type of bond representation	
	Valid values:	none wire tube
	Default value:	**wire**

Operands:
⟨atom1⟩ ⟨atom2⟩
The operands represent the numbers of the two atoms which define the bond to have its representation changed.

repdefault

Set representation of all atoms and bonds

Syntax:

repdefault *style*=wire | cpk | ballnstick | tube

Options:

style Default representation

Valid values: wire
cpk
ballnstick
tube

Default value: **wire**

replacefromhold

This command replaces the on-screen structure with a copy of the the structure from the specified hold set.

Syntax:

replacefromhold ⟨hold_name⟩

Operands:

⟨hold_name⟩

The name of the hold. This must be the name which was specified when the hold was created using the "hold" command.

repquick

Set representation of all atoms and bonds

Syntax:

repquick *default*=default | wire | cpk | ballnstick | tube
style=default | wire | cpk | ballnstick | tube [update]

Options:

default Default representation

	Valid values:	default wire cpk ballnstick tube
	Default value:	**wire**
style	Default representation	
	Valid values:	default wire cpk ballnstick tube
	Default value:	**default**

Operands:

[update]

When udpate is present the currently set representation will be applied to the on-screen structure.

resetcsearch

Deletes all the variables used in a conformational search.

Syntax:

resetcsearch

residuename

Set the residue name for all atoms which match the ASL specification.

Syntax:

residuename ⟨PDBNAME⟩ ⟨ASL⟩

Operands:

⟨PDBNAME⟩ ⟨ASL⟩

The first operand is the PDB residue name which will be used for all atoms which match the specification. The second operand is a valid ASL string which defines the set of atoms which are to have their residue names changed.

residuenumber

Set the residue number for all atoms which match the ASL specification.

Syntax:

residuenumber ⟨ res_num ⟩ ⟨ ASL ⟩

Operands:

⟨ res_num ⟩ ⟨ ASL ⟩

The first operand is an integer, optionally followed by a single-character insertion code, that specifies the residue number for all the atoms which match the ASL specification. If the trailing alphabetic character is omitted, the residue insertion code will be set blank. The second operand must be a valid ASL string which specifies all the atoms to have their residue number changed.

residuerenumber

Renumber the residues, starting with the starting number, for all residues which match the ASL specification.

Syntax:

residuerenumber ⟨ starting_res_num ⟩ ⟨ ASL ⟩

Operands:

⟨ starting_res_num ⟩ ⟨ ASL ⟩

The first operand is an integer which represents the starting residue number. The second operand must be a valid ASL string which specifies a set of residues to renumber.

restorepanels

Restores panel locations

Syntax:

restorepanels

retype

Change the atom type of the atom number specified by the operand to whatever type or element has previously been made current with the atom command.

Syntax:

retype ⟨atom_num⟩

Operands:

⟨atom_num⟩

An atom number representing the atom which is to have its type changed to the current type.

ribbon

Creates a new Ribbon.

Syntax:

ribbon *ambient*=⟨x⟩ *ambientback*=⟨x⟩ *calphalinewidth*=⟨n⟩ *calphatubesteps*=⟨n⟩ *calphatubewidth*=⟨x⟩ *color*=black | gray | darkblue | blue | lightblue | aquamarine | turquoise | springgreen | darkgreen | green | limegreen | yellowgreen | yellow | orange | maroon | red | pink | plum | purple | bluepurple | white *curvedlinesteps*=⟨n⟩ *curvedlinewidth*=⟨n⟩ *diffuse*=⟨x⟩ *diffuseback*=⟨x⟩ *display*=ribbonsonly | atomsonly | both *emission*=⟨x⟩ *emissionbackblue*=⟨x⟩ *emissionbackgreen*=⟨x⟩ *emissionbackred*=⟨x⟩ *helixcolor*=onecolor | twocolors *resolution*=high | low *ribbonendweight*=⟨x⟩ *ribbonhasthick*=yes | no *ribbonsteps*=⟨n⟩ *ribbonthick*=⟨x⟩ *ribbonweight*=⟨x⟩ *ribbonwidth*=⟨x⟩ *scheme*=constant | secondarystructure | chain | calphaatom | residuecharge | residueproperty | residuetype | residueposition | entry *shininess*=⟨x⟩ *shininessback*=⟨x⟩ *specular*=⟨x⟩ *specularback*=⟨x⟩ *sphereslices*=⟨n⟩ *spherestacks*=⟨n⟩ *strandarrowweight*=⟨x⟩ *strandarrowwidth*=⟨x⟩ *strandendweight*=⟨x⟩ *strandendweight1*=⟨x⟩ *strandsteps*=⟨n⟩ *strandthick*=⟨x⟩ *strandweight*=⟨x⟩ *strandwidth*=⟨x⟩ *style*=none | cartoon | ribbon | tube | thintube | curvedline | calphaline | calphatube *thintubesteps*=⟨n⟩ *thintubeweight*=⟨x⟩ *thintubewidth*=⟨x⟩ *tubesteps*=⟨n⟩ *tubeweight*=⟨x⟩ *tubewidth*=⟨x⟩ ⟨ASL-definition⟩

Options:

ambient
: Set material property - ambient, to its red, green, and blue components, for front face.

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0
Maximum:	1.0

ambientback
: Set material property - ambient, to its red, green, and blue components, for back face.

Valid values:	reals
Default value:	**0.2**
Minimum:	0.0
Maximum:	1.0

calphalinewidth
: Set linewidth for drawing CA Trace ribbons.

Valid values:	integers
Default value:	**2**

Minimum: 1
Maximum: 40

calphatubesteps

Set the steps of drawing ribbon CA Tube Trace from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values: integers
Default value: **3**
Minimum: 1
Maximum: 10

calphatubewidth

Set CA tube width for drawing CA Trace Tube ribbons.

Valid values: reals
Default value: **0.4**
Minimum: 0.001

color An option which controls the ribbon constant color.

Valid values: black
gray
darkblue
blue
lightblue
aquamarine
turquoise
springgreen
darkgreen
green
limegreen
yellowgreen
yellow
orange
maroon
red
pink
plum
purple
bluepurple
white
Default value: **green**

curvedlinesteps

Set the steps of drawing ribbon Curved Line from one node point (mapped from one CA atom) to another. It is used when the resolution option is HIGH.

Valid values: integers
Default value: **9**
Minimum: 1
Maximum: 100

curvedlinewidth
Set curve width for drawing Curved Line ribbons.

Valid values: integers
Default value: **2**
Minimum: 1
Maximum: 40

diffuse Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.4**
Minimum: 0.0
Maximum: 1.0

diffuseback
Set material property - diffuse, to its red, green, and blue components, for back face.

Valid values: reals
Default value: **0.2**
Minimum: 0.0
Maximum: 1.0

display An option which controls whether the atoms which define the ribbon are to be shown. The three options are: Ribbon Only, Atoms Only, and Both.

Valid values: ribbonsonly
atomsonly
both
Default value: **ribbonsonly**

emission Set material property - emission, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.05**
Minimum: 0.0
Maximum: 1.0

emissionbackblue
Set material property - emission, to its blue component, for back face. Emission R, G, B values are used to control the color of back face. The great the values, the lighter the back face.

Valid values: reals

Default value: **0.55**
Minimum: 0.0
Maximum: 1.0

emissionbackgreen
Set material property - emission, to its green component, for back face. Emission R, G, B values are used to control the color of back face. The great the values, the lighter the back face.

Valid values: reals
Default value: **0.52**
Minimum: 0.0
Maximum: 1.0

emissionbackred
Set material property - emission, to its red component, for back face. Emission R, G, B values are used to control the color of back face. The great the values, the lighter the back face.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

helixcolor An option which controls whether ribbons in the helical part are to be colored with a single color or with two colors and the inside of the helix has a contrasting color.

Valid values: onecolor
twocolors
Default value: **twocolors**

resolution Set the resolution for drawing ribbons.

Valid values: high
low
Default value: **high**

ribbonendweight
This weight value is used to control the radius of the Ribbon ends. The great the value, the small the radius.

Valid values: reals
Default value: **0.6**
Minimum: 0.001

ribbonhasthick
This bool value is used to determine which kind of ribbon will be drawn, a flat sheet or a solid ribbon has thick.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

ribbonsteps

Set the steps of drawing Ribbon ribbon from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values: integers
Default value: **3**
Minimum: 1
Maximum: 10

ribbonthick

This value is used to control the thick of ribbon, if the ribbon is display in solid ribbon.

Valid values: reals
Default value: **0.1**
Minimum: 0.02
Maximum: 2.0

ribbonweight

This weight value is used to control the radius of the rectangle cross-section of a strand ribbon. The great the value, the small the radius.

Valid values: reals
Default value: **3**
Minimum: 0.001

ribbonwidth

Set ribbon width for drawing Ribbon ribbons.

Valid values: reals
Default value: **1.3**
Minimum: 0.05
Maximum: 5.0

scheme An option which controls the ribbon coloring scheme.

Valid values: constant
secondarystructure
chain
calphaatom
residuecharge
residueproperty
residuetype
residueposition
entry
Default value: **secondarystructure**

shininess Set material property - shininess, for front face.

Valid values: reals

Default value: **80**
Minimum: 0.0
Maximum: 128.0

shininessback
Set material property - shininess, for back face.

Valid values: reals
Default value: **20**
Minimum: 0.0
Maximum: 128.0

specular Set material property - specular, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.7**
Minimum: 0.0
Maximum: 1.0

specularback
Set material property - specular, to its red, green, and blue components, for back face.

Valid values: reals
Default value: **0.2**
Minimum: 0.0
Maximum: 1.0

sphereslices
Set the slices of drawing sphere in the tube style. We use gluSphere() to draw a sphere. The higher the slices, the fine the sphere.

Valid values: integers
Default value: **18**
Minimum: 2

spherestacks
Set the stacks of drawing sphere in the tube style. We use gluSphere() to draw a sphere. The higher the stacks, the fine the sphere.

Valid values: integers
Default value: **18**
Minimum: 2

strandarrowweight
This weight value is used to control the radius of the strand arrow. The great the value, the small the radius.

Valid values: reals
Default value: **10**

Minimum: 0.001

strandarrowwidth

This ratio value is used to control the width of the strand arrow.

Valid values: reals
Default value: **2**
Minimum: 1.0

strandendweight

This weight value is used to control the radius of the strand ends. The great the value, the small the radius.

Valid values: reals
Default value: **0.6**
Minimum: 0.001

strandendweight1

This weight value is used to control the radius of the strand ends in Cartoon. The great the value, the small the radius.

Valid values: reals
Default value: **50**
Minimum: 0.001

strandsteps

Set the steps of drawing ribbon Strand from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values: integers
Default value: **3**
Minimum: 1
Maximum: 10

strandthick

Set strand thick for drawing Strand ribbons.

Valid values: reals
Default value: **0.2**
Minimum: 0.02
Maximum: 2.0

strandweight

Ribbons are defined with NURBS curce or surfaces that are defined by 4D homogeneous coordnate (x, y, z, w) control points array or mesh. The weight w can push/pull away/towards the curve/surface part near the control point, by decressing/increasing weight value. This weight value is used to control the radius of the rectangle cross-section of a strand ribbon. The great the value, the small the radius.

Valid values: reals

Default value: **5**
Minimum: 0.001

strandwidth

Set strandwidth for drawing Strand ribbons.

Valid values: reals
Default value: **1.5**
Minimum: 0.05
Maximum: 5.0

style An option which controls the ribbon style representation.

Valid values: none
cartoon
ribbon
tube
thintube
curvedline
calphaline
calphatube
Default value: **cartoon**

thintubesteps

Set the steps of drawing ribbon Thin Tube from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values: integers
Default value: **3**
Minimum: 1
Maximum: 10

thintubeweight

This weight value is used to control the radius of the Thin Tube ends. The great the value, the small the radius.

Valid values: reals
Default value: **1**
Minimum: 0.001

thintubewidth

Set thin tube width for drawing Thin Tube ribbons.

Valid values: reals
Default value: **0.4**
Minimum: 0.05
Maximum: 5.0

tubesteps Set the steps of drawing ribbon Tube from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values:	integers
Default value:	**3**
Minimum:	1
Maximum:	10

tubeweight This weight value is used to control the radius of the Tube ends. The great the value, the small the radius.

Valid values:	reals
Default value:	**1**
Minimum:	0.001

tubewidth Set tube width for drawing Tube ribbons.

Valid values:	reals
Default value:	**0.9**
Minimum:	0.05
Maximum:	5.0

Operands:

⟨ASL-definition⟩

The operand must be a valid string in the atom specification language. It will define which atoms are to have a ribbon drawn.

ribbondump

Print out the current option values of the ribbon command.

Syntax:

ribbondump

ringclosure

A command which defines a ring closure to be used during a conformational search.

Syntax:

ringclosure *maximum*=⟨ x ⟩ *minimum*=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Options:

maximum The maximum distance between the ends of the ring which will be accepted as a candidate for "closure".

Valid values: reals
Default value: **2.5**
Minimum: 0.0

minimum The minimum distance between the ends of the ring which will be accepted as a candidate for "closure".

Valid values: reals
Default value: **0.5**
Minimum: 0.0

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

The four atom numbers which define a point in a ring which is to be opened while new structures are generated in a conformational search. The actually opening takes place between the second and third atoms specified. Note that specifying a-b-c-d is the same as specifiying d-c-b-a.

rotate

Rotate in degrees whatever is specified in the transform set. This is either global (all atoms) or a local grouping defined via an ASL in the transform command.

Syntax:

rotate *x*=⟨ x ⟩ *y*=⟨ x ⟩ *z*=⟨ x ⟩ [reset]

Options:

x Amount in degrees to rotate in X

Valid values: reals
Default value: **0**

y Amount in degrees to rotate in Y

Valid values: reals
Default value: **0**

z Amount in degrees to rotate in Z
Valid values: reals
Default value: **0**

Operands:
[reset]
If reset is present the global rotation matrix will be reset.

run

This is a standard alias for **scriptrun** (see [scriptrun], page 383).

saveimage

Capture the current main structure window and save to an image file.

Syntax:

saveimage *format*=tiff | jpeg *jpeg_quality*=⟨ n ⟩ *png_compression*=⟨ n ⟩ *png_gamma*=⟨ x ⟩ *smooth*=yes | no ⟨ file_name ⟩

Options:

format Specifies the format of the saved image.
Valid values: tiff
jpeg
Default value: **tiff**

jpeg_quality
Quality of the JPEG file (1-100)
Valid values: integers
Default value: **75**

png_compression
Compression ratio of png format
Valid values: integers
Default value: **6**

png_gamma
Gamma of the image in png format

Valid values: reals
Default value: **2.2**

smooth Image saved will have smooth curves in it if enabled.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨file_name⟩

The file where the image will be saved.

saveimageheight

Sets the height of the image saved due to saveimage command KEY_OPERADN_SYNOPSIS: <height>

Syntax:

saveimageheight

saveimagewidth

Sets the width of the image saved due to saveimage command KEY_OPERADN_SYNOPSIS: <width>

Syntax:

saveimagewidth

savelayout

Save the size and position of all currently visible panels.

Syntax:

savelayout

scriptlogfile

Commands are by default logged to a temporary file which is deleted when the program terminates. If the "logfile" command is used to name the logfile then the commands will be logged to that file and that file will not be deleted when the program ends.

Syntax:

scriptlogfile *logging*=yes | no ⟨logfile_name⟩

Options:

logging If this option is set to "false" then no logging will be done to a file or in memory.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
⟨logfile_name⟩
The name of the file to which the commands will be logged. The full name of the file, including any suffix must be included.
Aliases:
logfile (see [logfile], page 205)

scriptrun

Run the script file whose name is given as the operands.

Syntax:

scriptrun ⟨script_name⟩

Operands:
⟨script_name⟩
The name of the script which is to be executed. The full name, including any suffix must be given. If the file is not in the local directory then a full pathname must be given.

Aliases:
run (see [run], page 381)

searchdbconfgen

Defines settings for Find Matches to Hypothesis Generate Conformers job.

Syntax:

searchdbconfgen *field*=mmffs *incorporate*=append | replace | ignore *maxdist*=⟨ x ⟩ *numsteps*=⟨ n ⟩ *postmaxiter*=⟨ n ⟩ *postprocessing*=yes | no *premaxiter*=⟨ n ⟩ *sampling*=standard | rapid | complete | thorough *solvation*=gbsa | distance_dependent *window*=⟨ x ⟩

Options:

field This determines which force field mmffs|mmff|opls2001 is used. Currently we always use mmffs, so it will have only one option value.

Valid values: mmffs
Default value: **mmffs**

incorporate
This option controls the incorporation of the results (replace or append).

Valid values: append
replace
ignore
Default value: **replace**

maxdist Maximum distance between atoms in equal structures.

Valid values: reals
Default value: **2**
Minimum: 0.0

numsteps An option which sets the number of steps which will be performed during the conformational search. This also limits number of conformations generated.

Valid values: integers
Default value: **100**
Minimum: 0

postmaxiter

This option determines the maximum number of iterations for post-minimization of generated structures.

Valid values:	integers
Default value:	**50**
Minimum:	0
Maximum:	9999999

postprocessing

Indicates whether or not to perform MacroModel postprocessing.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

premaxiter

This option determines the maximum number of iterations for pre-minimization of input structures.

Valid values:	integers
Default value:	**100**
Minimum:	0
Maximum:	9999999

sampling This determines whether rapid (standard) or thorough (complete) sampling will be used.

Valid values:	standard rapid complete thorough
Default value:	**standard**

solvation This determines whether GB/SA Water (gbsa) or Distance Dependent Dielectric (distance_dependent) solvation treatment is used.

Valid values:	gbsa distance_dependent
Default value:	**distance_dependent**

window The energy window (in kcal/mol) within which structures will be saved.

Valid values:	reals
Default value:	**10**
Minimum:	0.0

set

Creates a new named set. The set name must be a single token (or "quoted" if multiple tokens). A set can be redefined by specifying a new definition.

Syntax:

set ⟨set_name⟩ ⟨ASL-definition⟩

Operands:
⟨set_name⟩ ⟨ASL-definition⟩
The name which will be applied to the set. If the name contains embedded spaces then it must be enclosed in double quotation marks.

setread

Read set definitions from the file whose name is given as the operand. The file name usually has a ".set" suffix.

Syntax:

setread ⟨set_file_name⟩

Operands:
⟨set_file_name⟩
The name of the file from which the set definitions are to be read. The full name of the file (including any .set suffix) must be specified.

setwrite

Write the currently defined sets to the file whose name is given as the operand. The file name usually has a ".set" suffix

Syntax:

setwrite ⟨set_file_name⟩

Operands:
⟨set_file_name⟩

The name of the file to which the current set definitions are to be written . The full name of the file (including any .set suffix) must be specified.

showhwstereosetup

Write out the hardware stereo setup.

Syntax:

showhwstereosetup

showpanel

Show the panel whose name is given by the operands.

Syntax:

showpanel ⟨panel_name⟩ [:⟨tab_name⟩]

Operands:
⟨panel_name⟩ [:⟨tab_name⟩]
The first operand is the name of the panel which is to be displayed. The name must match to all characters. The names of the panels to be used in the "showpanel" command are displayed in parentheses after each item in the main menu bar. The optional second argument is the name of a tab folder within that panel which is to be made the current tab folder. The name of a tab folder to be used in the second optional operand is displayed in the associated panel. The name must match to all characters, but it is not case sensitive.

showpanels

Make visible panels previously hidden with HIDEPANELS

Syntax:

showpanels

sitemap

Sets the parameters for sitemap

Syntax:

sitemap *boxmargin*=⟨x⟩ *boxtype*=box | ligand *cutoff*=⟨x⟩ *cutoffscheme*=atom | neutral *gridspacing*=standard | high *incorporate*=append | replace | ignore *method*=h2o | dipole *source*=selected_entries | workspace | file

Options:

boxmargin This value is used to increase the size of the box.

Valid values: reals
Default value: **6**
Minimum: 0.

boxtype This option indicates whether to treat the box ASL as defining a box or as defining a ligand.

Valid values: box
ligand
Default value: **box**

cutoff This specifies the cutoff value to use for the sitemap calculation, in angstroms.

Valid values: reals
Default value: **20**
Minimum: 0.

cutoffscheme
This option controls the cutoff scheme.

Valid values: atom
neutral
Default value: **atom**

gridspacing
This option indicates what type of grid spacing (standard or high) to use for the sitemap.

Valid values: standard
high
Default value: **standard**

incorporate
This option controls the incorporation of the results.

Valid values:	append replace ignore
Default value:	**replace**

method This option controls the method.

Valid values:	h2o dipole
Default value:	**dipole**

source This is the source for the structure.

Valid values:	selected_entries workspace file
Default value:	**workspace**

sitemapbox

Sets the box parameters for sitemap

Syntax:

sitemapbox ⟨ASL⟩

Operands:
⟨ASL⟩
The ASL defines the box to use for the sitemap calculation.

sitemapset

This command is used to specify the atoms for which the sitemap calculation will be performed.

Syntax:

sitemapset ⟨ASL⟩

Operands:
⟨ASL⟩
The ASL defines the set of atoms for which the sitemap calculation is to be performed.

sitemapstart

Starts a sitemap job with the current parameters.

Syntax:

sitemapstart

sitemapwrite

Writes the sitemap input file with the current parameters.

Syntax:

sitemapwrite

sleep

This is a standard alias for **energysleep** (see [energysleep], page 88).

specifiedname

Set the specified name to that specified for all atoms which match the ASL specification.

Syntax:

specifiedname ⟨specified_name⟩ ⟨ASL⟩

Operands:

⟨specified_name⟩ ⟨ASL⟩

The first operand is the specified name which is to be applied to the atom. Only the first 20 characters of the specified name will be used. The second operand is the ASL specification for all the atoms which are to have the specified name applied.

spotcenter

Center the given atom on the screen. Make it the center of global rotation.

Syntax:

spotcenter ⟨atom_num⟩

Operands:

⟨atom_num⟩

The number of the atom which is to be centered on the screen and to become the center of rotation.

stop

This is a standard alias for **energystop** (see [energystop], page 88).

strikebuildqsar

Runs a Build QSAR job.

Syntax:

strikebuildqsar *activity_property*=⟨text⟩ *max_pls_factors*=⟨n⟩ *method*=pls | pca | mlr *remove_outliers*=yes | no *which_descriptors*=all | subset ⟨job name⟩

Options:

activity_property
The name of the activity property.

Valid values: text strings
Default value:

max_pls_factors
How many factors for Partial Least Squares.

Valid values: integers
Default value: **1**
Minimum: 0

method The regression method to use.

Valid values: pls
pca
mlr
Default value: **pls**

remove_outliers
Set to true to automatically remove outliers.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

which_descriptors
Which descriptors to use.

Valid values: all
subset
Default value: **all**

Operands:

⟨job name⟩
The name of the job to run.

strikedeletemodel

Deletes the given model.

Syntax:

strikedeletemodel ⟨model name⟩

Operands:
⟨model name⟩
The name of the model to delete.

strikeexportmodel

Exports a model from Strike.

Syntax:

strikeexportmodel ⟨file⟩

Operands:
⟨file⟩
The file to export the Strike model to.

strikeextendselectdescriptor

Extends the selected descriptors via the given descriptor.

Syntax:

strikeextendselectdescriptor ⟨M2IO descriptor name⟩

Operands:
⟨M2IO descriptor name⟩
The M2IO data name of the descriptor to select.

strikeimportmodel

Imports a model to the Strike panels.

Syntax:

strikeimportmodel ⟨file⟩

Operands:
⟨file⟩
The file to import the Strike model from.

strikepredict

Runs a prediction job.

Syntax:

strikepredict ⟨job name⟩

Operands:
⟨job name⟩
The name of the job to run.

strikeselectdescriptor

Selects the given descriptor

Syntax:

strikeselectdescriptor ⟨M2IO descriptor name⟩

Operands:
⟨M2IO descriptor name⟩
The M2IO data name of the descriptor to select.

strikeselectmodel

Selects only the given model.

Syntax:

strikeselectmodel ⟨model name⟩

Operands:
⟨model name⟩
The name of the model to select.

strikesimilarity

Runs a similarity job.

Syntax:

strikesimilarity *job_type*=atompairs | descriptors ⟨job name⟩

Options:

job_type	Whether to calculate atom pair or descriptor similarities.	
	Valid values:	atompairs descriptors
	Default value:	**atompairs**

Operands:
⟨job name⟩
The name of the job to run.

striketoggleselectdescriptor

Toggles the selection of the given descriptor on or off.

Syntax:

striketoggleselectdescriptor ⟨M2IO descriptor name⟩

Operands:
⟨M2IO descriptor name⟩
The M2IO data name of the descriptor to select.

structalignatoms

Sets the ASL that the next struct align job will operate on.

Syntax:

structalignatoms ⟨ASL⟩

Operands:
⟨ASL⟩
The residues to align.

structalignstart

Start a struct align job with the current settings.

Syntax:

structalignstart

substructure

Specifies a set of atoms to be used as the "substructure" during a substructure minimization or dynamics simulation.

Syntax:

substructure *fillres*=yes | no *radius*=⟨x⟩ ⟨ASL⟩

Options:

fillres A boolean option that determines if the substructure definition will be expended to complete residue boundaries.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

radius The radius from the basic substructure definition within which atoms will be included in the substructure.
Valid values: reals
Default value: **0**
Minimum: 0.0

Operands:
⟨ASL⟩
A string in the atom specification language. All atoms which match this will be treated as part of the substructure in a substructure minimization.

substructurefile

This is a standard alias for **substructurefilewrite** (see [substructurefilewrite], page 397).

substructurefileread

Will read a .sbc file and replace the current substructure and constrained/fixed atom shells.

Syntax:

substructurefileread ⟨sbc_file_name⟩

Operands:
⟨sbc_file_name⟩
The complete name (including the *.sbc suffix) of the file from which the substructure information will be read.

substructurefilewrite

Will write a .sbc file with the current substructure and constrained/fixed atom shells.

Syntax:

substructurefilewrite ⟨sbc_file_name⟩

Operands:
⟨sbc_file_name⟩
The complete name (including the *.sbc suffix) of the file to which the substructure information will be written.

Aliases:
substructurefile (see [substructurefile], page 397)

substructureshell

Specifies a set of atoms forming a "shell" around the substructure defined by the "subs" command.

Syntax:

substructureshell *addatoms*=⟨text⟩ *constant*=⟨x⟩
fillres=yes | no *frozen*=yes | no *radius*=⟨x⟩ ⟨shell_number⟩

Options:

addatoms A string in the atom specification language. All atoms which match this will be treated as part of the shell.
Valid values: text strings
Default value:

constant The harmonic force constant to be applied to the constrained shell.
Valid values: reals
Default value: **200**
Minimum: 0.0

fillres A boolean option which determines whether the shell is to be made up of complete residues.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

frozen A boolean option which determines whether the atoms in the shell are to be "frozen" during the substructure calculation.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

radius The radius of a "shell" of atoms around the "substructure" during a substructure energy procedure.
Valid values: reals
Default value: **0**
Minimum: 0.0

Operands:

⟨ shell_number ⟩
The number of the shell. Shells are usually numbered from 1, but any sequence will work. The shell with the lowest number is defined relative to the atom in the substructure. Subsequent shells are defined relative to next lowest numbered shell.

superimpose

Perform a superposition using previously defined atoms or all corresponding atoms in all onscreen entries if all is used.

Syntax:

superimpose *inplace*=yes | no [all]

Options:

inplace Enable superposition to be performed in place (effectively just calculates the RMS and doesn't actually move any entries.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
[all]
If all is present then an attempt will be made to superimpose all onscreen entries. Otherwise only the atom pairs defined by superimposeatom commands will be used to perform a superposition.

superimposeatom

Define an atom pair for which will be superimposed in a subsequent superposition operation.

Syntax:

superimposeatom ⟨ atom1 ⟩ ⟨ atom2 ⟩

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩

Two atom numbers which represent an atom pair to be superimposed by a subsequent superimpose command. The two atoms must be from different entries and all superimposeatom commands must specify entries in the same order.

superimposeset

Use an ASL expression to define superposition atom pairs. The ASL set must define exactly the same number of atoms in each on-screen entry.

Syntax:

superimposeset ⟨ASL⟩

Operands:

⟨ASL⟩

A string in the atom specification language. This set must define exactly the same number of atoms in each on-screen entry and those atoms will become the basis for superposition.

surfacedelete

Deletes the given surface.

Syntax:

surfacedelete *entry*=⟨text⟩ ⟨surface⟩

Options:

entry The entry that the surface belongs to.

Valid values: text strings
Default value:

Operands:

⟨surface⟩

The name of the surface to delete.

surfacedisplay

Displays or undisplays the given surface

Syntax:

surfacedisplay *display*=yes | no *entry*=⟨ text ⟩ ⟨ surface ⟩

Options:

display Sets whether or not to display the surface.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

entry The entry that the surface belongs to.
Valid values: text strings
Default value:

Operands:
⟨ surface ⟩
The name of the surface to display.

surfaceduplicate

Duplicates the given surface.

Syntax:

surfaceduplicate *entry*=⟨ text ⟩ ⟨ surface ⟩

Options:

entry The entry that the surface belongs to.
Valid values: text strings
Default value:

Operands:
⟨ surface ⟩
The name of the surface to duplicate.

surfaceextended

Creates a new extended radius surface for the current workspace.

Syntax:

surfaceextended *context*=entry | molecule | workspace | none | asl *grid_spacing*=⟨x⟩ *probe_radius*=⟨x⟩ *transparency*=⟨x⟩
name of the surface, followed by ASL defining the atoms to be surfaced

Options:

context
This option sets the context for generating a surface.

Valid values:	entry molecule workspace none asl
Default value:	**entry**

grid_spacing
This option sets the grid spacing for the surface.

Valid values:	reals
Default value:	**0.3**
Minimum:	0.01

probe_radius
This is the probe radius.

Valid values:	reals
Default value:	**1.4**
Minimum:	1.0

transparency
The transparency option controls how transparent the surface appears. A value of 100 means 100 percent transparent. 0 means completely opaque.

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	100.0

Operands:
name of the surface, followed by ASL defining the atoms to be surfaced
Name of the extended radius surface to be created

surfaceextendedradiuscontext

Defines a set of atoms for which a surface can be clipped against with the surfaceextendedradius command.

Syntax:

surfaceextendedradiuscontext ⟨ASL⟩

Operands:
⟨ASL⟩
The ASL expression which defines the atoms the surface will be clipped against.

surfaceextendedradiusset

Defines a set of atoms for which a surface can be created for with the surfaceextendedradius command.

Syntax:

surfaceextendedradiusset ⟨ASL⟩

Operands:
⟨ASL⟩
The ASL expression which defines the atoms the surface will be created for.

surfacemolecular

Creates a new molecular surface for the current workspace.

Syntax:

surfacemolecular *context*=entry | molecule | workspace | none | asl *grid_spacing*=⟨x⟩ *probe_radius*=⟨x⟩ *transparency*=⟨x⟩ ⟨surface name⟩

Options:

context
This option sets the context for generating a surface.

Valid values:	entry molecule workspace none asl
Default value:	**entry**

grid_spacing
This option sets the grid spacing for the surface.

Valid values:	reals
Default value:	**0.8**
Minimum:	0.01

probe_radius
This is the probe radius.

Valid values:	reals
Default value:	**1.4**
Minimum:	1.0

transparency
The transparency option controls how transparent the surface appears. A value of 100 means 100 percent transparent. 0 means completely opaque.

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	100.0

Operands:

⟨surface name⟩

Name of the molecular surface to be created

surfacemolecularcontext

Defines a set of atoms for which a surface will be clipped against with the surfacemolecular command.

Syntax:

surfacemolecularcontext ⟨ASL⟩

Operands:

⟨ASL⟩

The ASL expression which defines the atoms the surface will be clipped against.

surfacemolecularset

Defines a set of atoms for which a surface can be created for with the surfacemolecular command.

Syntax:

surfacemolecularset ⟨ASL⟩

Operands:

⟨ASL⟩

The ASL expression which defines the atoms the surface will be created for.

surfacerename

Renames the given surface.

Syntax:

surfacerename *entry*=⟨text⟩ *newname*=⟨text⟩ ⟨surface⟩

Options:

entry The entry that the surface belongs to.

Valid values: text strings
Default value:

newname The new name for the surface.

Valid values: text strings
Default value:

Operands:
⟨ surface ⟩
The name of the surface to rename.

surfacescheme

Sets the color scheme for the given surface.

Syntax:

surfacescheme *color*=⟨ text ⟩ *colorramp*=⟨ text ⟩
datarange=on_surface | entire_volume *entry*=⟨ text ⟩
negativecolor=⟨ text ⟩ *scheme*=⟨ text ⟩ *schemevolume*=⟨ text ⟩
⟨ surface ⟩

Options:

color Sets the color for a constant color.

Valid values: text strings
Default value: **white**

colorramp Sets the color ramp for the surface when color scheme is Map Values From Volume.

Valid values: text strings
Default value: **redwhiteblue**

datarange Sets the data range to which we apply color ramp.

Valid values: on_surface
entire_volume
Default value: **on_surface**

entry The entry that the surface belongs to.

Valid values: text strings
Default value:

negativecolor
Sets the color for the negative surface.

Valid values: text strings
Default value: **white**

scheme Sets the color scheme.

Valid values: text strings
Default value:

schemevolume
Sets the color scheme volume name.

Valid values: text strings
Default value:

Operands:
⟨ surface ⟩
The name of the surface to set the color scheme for.

surfacesetcomment

Changes the comments of a given surface.

Syntax:

surfacesetcomment *comment*=⟨ text ⟩ *entry*=⟨ text ⟩ ⟨ surface ⟩

Options:

comment The new comments for the surface.

Valid values: text strings
Default value:

entry The entry that the surface belongs to.

Valid values: text strings
Default value:

Operands:
⟨ surface ⟩
The name of the surface to change the comments for.

surfacesetisovalue

Changes the isovalue of a given surface.

Syntax:

surfacesetisovalue *entry*=⟨text⟩ *isovalue*=⟨x⟩ ⟨surface⟩

Options:

entry The entry that the surface belongs to.
Valid values: text strings
Default value:

isovalue The new isovalue for the surface.
Valid values: reals
Default value: **0.3**

Operands:
⟨surface⟩
The name of the surface to change the isovalue for.

surfacesetviewasl

This command sets a surface to only display the portions of the surface within a given distance of a given ASL.

Syntax:

surfacesetviewasl *distance*=⟨x⟩ *entry*=⟨text⟩ *surface*=⟨text⟩ *use*=yes | no ASL

Options:

distance The distance to the ASL.
Valid values: reals
Default value: **5**

entry The entry that the surface belongs to.
Valid values: text strings
Default value:

surface The name of the surface to operate on.
Valid values: text strings
Default value:

use Determines whether or not to use the view by ASL.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
ASL
The asl which controls the visible surface.

surfacestyle

Sets the drawing style for the given surface.

Syntax:

surfacestyle *entry*=⟨text⟩ *style*=solid | mesh | dots ⟨surface⟩

Options:

entry The entry that the surface belongs to.
Valid values: text strings
Default value:

style Sets the drawing style.
Valid values: solid
mesh
dots
Default value: **solid**

Operands:
⟨surface⟩
The name of the surface to set the style for.

surfacetransparency

Sets the transparency for the given surface.

Syntax:

surfacetransparency *entry*=⟨text⟩ *transparency*=⟨x⟩ ⟨surface⟩

Options:

entry The entry that the surface belongs to.
Valid values: text strings

Default value:

transparency
Sets the transparency percentage.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 100.0

Operands:
⟨ surface ⟩
The name of the surface to set transparency on.

surfacevdw

Creates a new vdW surface for the current workspace.

Syntax:

surfacevdw *asl*=⟨ text ⟩ *context*=entry | molecule | workspace | none | asl *grid_spacing*=⟨ x ⟩ *transparency*=⟨ x ⟩ Name of the surface

Options:

asl
This option sets the ASL specification for the atoms which define the surface.

Valid values: text strings
Default value:

context
This option sets the context for generating a surface.

Valid values: entry
molecule
workspace
none
asl
Default value: **entry**

grid_spacing
This option sets the grid spacing for the surface.

Valid values: reals
Default value: **0.2**
Minimum: 0.01

transparency

The transparency option controls how transparent the surface appears. A value of 100 means 100 percent transparent. 0 means completely opaque.

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	100.0

Operands:
Name of the surface
Name of the vdW surface to be created

surfacevdwcontext

Defines a set of atoms for which a surface will be clipped against with the surfacevdw command.

Syntax:

surfacevdwcontext ⟨ASL⟩

Operands:
⟨ASL⟩
The ASL expression which defines the atoms the surface will be clipped against.

surfacevdwset

Defines a set of atoms for which a surface can be created for with the surfacevdw command.

Syntax:

surfacevdwset ⟨ASL⟩

Operands:
⟨ASL⟩

The ASL expression which defines the atoms the surface will be created for.

surfaceviewaslset

Defines a set of atoms for the View by ASL property which will be used by the surfacesetviewasl command.

Syntax:

surfaceviewaslset ⟨ASL⟩

Operands:

⟨ASL⟩

The ASL expression which defines the atoms that will be used for the View by ASL property.

symmetrizeworkspace

Symmetrizes the workspace finding the point groups.

Syntax:

symmetrizeworkspace *tolerance*=⟨x⟩ analyze|update

Options:

tolerance Tolerance for finding the pointing groups and symmetrizing the workspace

Valid values:	reals
Default value:	**0.04**
Minimum:	0.04
Maximum:	1.00

Operands:

analyze|update

If analyze, finds the point groups for the current tolerance. If update, symmetrize the Workspace with the current tolerance.

system

Execute a command from the system.

Syntax:

system ⟨command⟩

Operands:
⟨command⟩
A command which is to be executed from the current shell.

tableresizecolumn

Set the width of the specified column in the current table.

Syntax:

tableresizecolumn ⟨columnname⟩ ⟨width⟩

Operands:
⟨columnname⟩ ⟨width⟩
The name of the column to resize. This is the name displayed in the column header. The width to set the column to. This is the new width for the column.

tablesort

Sort current entry selection in the specified table.

Syntax:

tablesort *field*=⟨text⟩ *order*=ascending | descending ⟨table⟩

Options:

field The name of the property to be sorted on.
Valid values: text strings
Default value: **s_m_entry_name**

order This option sets order of sorted values to be either ascending or descending.

Valid values: ascending
descending
Default value: **ascending**

Operands:
⟨ table ⟩
The name of the table to sort.

tablesortall

Sort all rows in the specified table.

Syntax:

tablesortall ⟨ table ⟩

Operands:
⟨ table ⟩
The name of the table to sort.

tablesortfields

Sets multiple fields for sorting.

Syntax:

tablesortfields ⟨ field ⟩ ⟨ ascending | descending ⟩

Operands:
⟨ field ⟩ ⟨ ascending | descending ⟩
The fields to sort together with the sort order: ascending or descending.

tableunselectnonsubset

Unselects all entries which are not in the current table's subset.

Syntax:

tableunselectnonsubset

tile

Spread the on-screen entries out in a tile-pattern.

Syntax:

tile

toolbarmain

Used to control the main toolbar display features.

Syntax:

toolbarmain *location*=left | right

Options:

location Determines main toolbar's location in main drawing window, 1=left and 2=right.

Valid values: left
right
Default value: **left**

torsioncheck

Specifies four atoms which define a torsion to be checked during a conformational search.

Syntax:

torsioncheck *maximum*=⟨x⟩ *minimum*=⟨x⟩ ⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩

Options:

maximum The maximum allowed value for the torsion. during the torsion check.

Valid values: reals
Default value: **180**
Minimum: 0.0

minimum The minimum allowed value for the torsion check (Degrees)

Valid values: reals
Default value: **0**

Operands:

⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩

The numbers of four atoms which define a torsion angle to be checked during the conformational search. Note that specifying a-b-c-d is the same as specifying d-c-b-a.

torsiongroup

Sets the current torsion group and the conformation within that group

Syntax:

torsiongroup ⟨group_name⟩ ⟨conformation_name⟩

Operands:

⟨group_name⟩ ⟨conformation_name⟩

The first operand must be the name of a torsion group within the current fragment mode. The second operand must be the name of a conformation within that torsion group.

transform

Specify what is to be transformed.

Syntax:

transform *centerlocal*=centroid | atom *gui*=none | rotate | translate *rsensitivity*=⟨ x ⟩ *scope*=global | local *threshhold*=⟨ n ⟩ *tsensitivity*=⟨ x ⟩ [reset]

Options:

centerlocal

Whether local center is an atom or centroid of group of atoms

Valid values: centroid
atom
Default value: **centroid**

gui

Whether rotation, translation or nothing is performed

Valid values: none
rotate
translate
Default value: **rotate**

rsensitivity

Mouse rotation sensitivity. Larger is more sensitive.

Valid values: reals
Default value: **75**
Minimum: 1.0
Maximum: 500.0

scope

Whether global or local transformations are done

Valid values: global
local
Default value: **global**

threshhold

How many pixels a mouse move event must contain before any transformation will occur

Valid values: integers
Default value: **1**
Minimum: 1
Maximum: 30

tsensitivity

[NOTE: This option is no longer used.] Mouse translation sensitivity. Larger is more sensitive.

Valid values: reals
Default value: **3.33**
Minimum: 1.0
Maximum: 500.0

Operands:
[reset]
Specifies that the transformations are to be reset - this must be reset .

translate

Translate in Angstroms whatever is specified in the transform set. This is either global (all atoms) or a local grouping defined useing an ASL expression in the transform command.

Syntax:

translate *x*=⟨ x ⟩ *y*=⟨ x ⟩ *z*=⟨ x ⟩

Options:

x Amount in Angstroms to translate in X
Valid values: reals
Default value: **0**

y Amount in Angstroms to translate in Y
Valid values: reals
Default value: **0**

z Amount in Angstroms to translate in Z
Valid values: reals
Default value: **0**

undisplayatom

Undisplay atoms in the set described by the ASL.

Syntax:

undisplayatom ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language which describes the set of atoms which are to be undisplayed.

undisplaypose

This command removes the pose set structure from the workspace in the poseview.

Syntax:

undisplaypose ⟨pose_name⟩

Operands:
⟨pose_name⟩
The name of the pose structure.

undo

Undo the effect of the last change on the on-screen structure.

Syntax:

undo

unhookimport

Restores normal Import behavior.

Syntax:

unhookimport

uniquename

With the set of specified atoms, make the atom name unique. This is done by adding a "~N" to each duplicate name where the "N" is a digit which represents how often this name is repeated in the set.

Syntax:

uniquename

uniquepdb

Set unique PDB atom names (by residue) for all atoms that match the ASL specification.

Syntax:

uniquepdb ⟨ASL⟩

Operands:

⟨ASL⟩

The operand is a valid ASL string that defines the set of atoms that are to have their PDB atom names changed.

update

This is a standard alias for **energyupdate** (see [energyupdate], page 89).

updateribbons

Update existing ribbons with current ribbon style and color scheme.

Syntax:

updateribbons

varymolecule

A command which defines a rotatable/translatable molecule during a conformational search. The molecule is defined by specifying any atom which belongs to it.

Syntax:

varymolecule *rmax*=⟨ x ⟩ *rmin*=⟨ x ⟩ *tmax*=⟨ x ⟩ *tmin*=⟨ x ⟩
⟨ atom_number ⟩

Options:

rmax The maximum value for the molecule rotation.

Valid values:	reals
Default value:	**180**
Minimum:	0.0

rmin The minimum value for the molecule rotation.

Valid values:	reals
Default value:	**0**
Minimum:	0.0

tmax The maximum value for the molecule translation

Valid values:	reals
Default value:	**1**
Minimum:	0.0

tmin The minimum value for the molecule translation

Valid values:	reals
Default value:	**0**
Minimum:	0.0

Operands:

⟨ atom_number ⟩

The number of an atom which is a member of the molecule which is to be translated or rotated.

varytorsion

Defines a rotatable bond to be used in a conformational search.

Syntax:

varytorsion *maximum*=⟨ x ⟩ *minimum*=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩

Options:

maximum The maximum value for the torsional rotation.

Valid values:	reals
Default value:	**180**
Minimum:	0.0

minimum The minimum value for the torsional rotation.

Valid values:	reals
Default value:	**0**
Minimum:	0.0

Operands:

⟨atom1⟩ ⟨atom2⟩

The two atom numbers which define a bond to be rotated in a conformational search. These two atoms must have a bond (usually single) between them. Note that specifying a-b is the same as specifying b-a.

viewmatrix

Sets workspace rotation matrix to the matrix supplied When 'inverse' is supplied, it sets the inverse roation matrix and when 'nocenter', No center rotation matrix. And in the absence of both the keywords, it sets the rotation matrix.

Syntax:

viewmatrix

viewposecontacts

This command alters contact settings, to display bad and ugly receptor/ligand contacts in the pose viewer.

Syntax:

viewposecontacts

viewposehbonds

This command alters hbond settings, to display receptor/ligand H-bonds in the pose viewer.

Syntax:

viewposehbonds

viewreset

Resets the viewing transform to original.

Syntax:

viewreset

viewrestore

Sets the current viewing transform to be the saved or "home" transform.

Syntax:

viewrestore

viewsave

Saves the current viewing transform.

Syntax:

viewsave

viewvolume

Sets workspace view bounding box to the supplied values

Syntax:

viewvolume *bottom*=⟨x⟩ *far*=⟨x⟩ *left*=⟨x⟩ *near*=⟨x⟩ *right*=⟨x⟩ *top*=⟨x⟩

Options:

bottom Bottom co-ordinate of the bounding box
Valid values: reals
Default value: **-5**

far Far co-ordinate of the bounding box
Valid values: reals
Default value: **-10**

left Left co-ordinate of the bounding box
Valid values: reals
Default value: **-5**

near Near co-ordinate of the bounding box
Valid values: reals
Default value: **10**

right Right co-ordinate of the bounding box
Valid values: reals
Default value: **5**

top Top co-ordinate of the bounding box
Valid values: reals
Default value: **5**

visimport

Creates all volumes and surfaces in the given file.

Syntax:

visimport *entry*=⟨text⟩ *isovalue*=⟨x⟩ *transparency*=⟨x⟩
⟨file.vis⟩

Options:

entry This is the entry to associate the objects with.

Valid values: text strings
Default value:

isovalue If any of the volumes do not have associated surfaces or suggested isovalues, then this isovalue will be used to isosurface the volumes.

Valid values: reals
Default value: **0.03**

transparency
This option sets the transparency of any surfaces.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 100.0

Operands:
⟨file.vis⟩
The name of the visio or Jaguar plot file to load.

volumedelete

Deletes the given volume.

Syntax:

volumedelete *entry*=⟨text⟩ ⟨volume⟩

Options:

entry The entry that the volume belongs to.

Valid values: text strings
Default value:

Operands:
⟨ volume ⟩
The name of the volume to delete.

volumeisosurface

Isosurfaces the given volume.

Syntax:

volumeisosurface *entry*=⟨ text ⟩ *isovalue*=⟨ x ⟩ *surface*=⟨ text ⟩
⟨ volume ⟩

Options:

entry The entry that the volume belongs to.
Valid values: text strings
Default value:

isovalue The isovalue to use for the isosurfacing.
Valid values: reals
Default value: **0.1**

surface The name for the new surface.
Valid values: text strings
Default value:

Operands:
⟨ volume ⟩
The name of the volume to isosurface.

volumerename

Renames the given volume.

Syntax:

volumerename *entry*=⟨ text ⟩ *newname*=⟨ text ⟩ ⟨ volume ⟩

Options:

entry The entry that the volume belongs to.

Valid values: text strings
Default value:

newname The new name for the volume.

Valid values: text strings
Default value:

Operands:
⟨ volume ⟩
The name of the volume to rename.

wake

This is a standard alias for **energywake** (see [energywake], page 90).

workspaceselectionadd

Extends workspace selection with the atoms confirming to the supplied ASL.

Syntax:

workspaceselectionadd ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
ASL representing the set of atoms

workspaceselectionclear

Resets current workspace selection to none.

Syntax:

workspaceselectionclear

workspaceselectioninvert

Flips the state of an atom in the workspace selection. i.e., removes if the atom already exists or adds if it doesn't.

Syntax:

workspaceselectioninvert ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
ASL representing the set of atoms

workspaceselectionreplace

Overwrites the workspace selection with the supplied set of atoms.

Syntax:

workspaceselectionreplace ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
ASL representing the set of atoms

workspaceselectionsubtract

Removes atoms confirming to the supplied ASL from the workspace selection.

Syntax:

workspaceselectionsubtract ⟨ASL⟩

Operands:
⟨ASL⟩
ASL representing the set of atoms

write

This is a standard alias for **filewrite** (see [filewrite], page 118).

writeangle

Write the angles in the angle table to a file.

Syntax:

writeangle *atomnumber*=entry | molecule | workspace
delimiter=comma | tab | userdefined *userdelimiter*=⟨text⟩
⟨file_name⟩

Options:

atomnumber
This option is deprecated.
Valid values: entry
molecule
workspace
Default value: **entry**

delimiter This option sets what delimiter will be used in the output file: comma=1, tab=2, and userdefined=3.
Valid values: comma
tab
userdefined
Default value: **comma**

userdelimiter
The delimiter string defined by users. It is valid only when option delimiter is userdefined.
Valid values: text strings
Default value:

Operands:

⟨ file_name ⟩

The name of the file to which the angles will be written.

writecontact

Write Contacts information to a file.

Syntax:

writecontact *atomnumber*=entry | molecule | workspace
delimiter=comma | tab | userdefined *userdelimiter*=⟨ text ⟩
⟨ file_name ⟩

Options:

atomnumber
This option is deprecated.

Valid values: entry
molecule
workspace
Default value: **entry**

delimiter This option sets what delimiter will be used in the output file: comma=1, tab=2, and userdefined=3.

Valid values: comma
tab
userdefined
Default value: **comma**

userdelimiter
The delimiter string defined by users. It is valid only when option delimiter is userdefined.

Valid values: text strings
Default value:

Operands:

⟨ file_name ⟩

The name of the file to which Contacts information will be written.

writedihedral

Write the dihedrals in the dihedral table to a file.

Syntax:

writedihedral *atomnumber*=entry | molecule | workspace
delimiter=comma | tab | userdefined *userdelimiter*=⟨ text ⟩
⟨ file_name ⟩

Options:

atomnumber
This option has been deprecated.

Valid values: entry
molecule
workspace
Default value: **entry**

delimiter This option sets what delimiter will be used in the output file: comma=1, tab=2, and userdefined=3.

Valid values: comma
tab
userdefined
Default value: **comma**

userdelimiter
The delimiter string defined by users. It is valid only when option delimiter is userdefined.

Valid values: text strings
Default value:

Operands:
⟨ file_name ⟩
The name of the file to which the dihedrals will be written.

writedistance

Write the distances in the distance table to a file.

Syntax:

writedistance *atomnumber*=entry | molecule | workspace
delimiter=comma | tab | userdefined *userdelimiter*=⟨ text ⟩
⟨ file_name ⟩

Options:

atomnumber
This option is deprecated.
Valid values: entry
molecule
workspace
Default value: **entry**

delimiter This option sets what delimiter will be used in the output file: comma=1, tab=2, and userdefined=3.
Valid values: comma
tab
userdefined
Default value: **comma**

userdelimiter
The delimiter string defined by users. It is valid only when option delimiter is userdefined.
Valid values: text strings
Default value:

Operands:
⟨ file_name ⟩
The name of the file to which the distances will be written.

writehbond

Write H-Bonds information to a file.

Syntax:

writehbond *atomnumber*=entry | molecule | workspace
delimiter=comma | tab | userdefined *userdelimiter*=⟨ text ⟩
⟨ file_name ⟩

Options:

atomnumber
This option is deprecated.

Valid values: entry
molecule
workspace
Default value: **entry**

delimiter This option sets what delimiter will be used in the output file: comma=1, tab=2, and userdefined=3.

Valid values: comma
tab
userdefined
Default value: **comma**

userdelimiter
The delimiter string defined by users. It is valid only when option delimiter is userdefined.

Valid values: text strings
Default value:

Operands:

⟨file_name⟩

The name of the file to which H-Bonds information will be written.

writeposefile

If the write command is issued with options, but without a filename then nothing is written but the options are updated. If the write command alone is issued then writing is performed to the currently open file with the current options. A write command with both options and a file name specified will result in the write being performed with the new options to the specified file.

Syntax:

writeposefile *append*=yes | no ⟨file_name⟩

Options:

append This option determines whether to append to the file which is going to be written.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨file_name⟩

The name of the file to which the structure will be written from the pose-viewer.

xcluster

This keyword is used to set various options associated with starting XCluster jobs from Maestro.

Syntax:

xcluster *cluster_by*=atom | torsion *compare_enantiomers*=yes | no *input_file*=⟨text⟩ *job*=⟨text⟩ *rms_in_place*=yes | no *structure_source*=selected_entries | workspace | file

Options:

cluster_by Whether to use atomic RMS or torsional RMS differences for clustering analysis.

Valid values: atom
torsion
Default value: **atom**

compare_enantiomers
A boolean which controls whether XCluster will perform superposition in such a way so as to compare enantiomers.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

input_file The name of the structure input file.

Valid values: text strings
Default value:

job The name for the XCluster job.

Valid values: text strings
Default value: **xclustmp**

rms_in_place
A boolean which controls whether XCluster will calculate RMS differences without first doing a superposition.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

structure_source
: Whether to use the selected entries in the current project, or a specified file with multiple structures as structure input for the job.

Valid values:	selected_entries workspace file
Default value:	**selected_entries**

xclusterstart

Start a XCluster job with the current settings.

Syntax:

xclusterstart

xclusterwrite

Write a XCluster input file with the current settings.

Syntax:

xclusterwrite

zoom

Zoom in or out (i.e. scale up or down)

Syntax:

zoom *factor*=⟨ x ⟩ in | out

Options:

factor
: A zooming (scaling) factor which determines by how much the on-screen structures is scaled. 1.0 is 1 percent, 99.0 is 99 percent, etc. Values are used relative to current size

Valid values:	reals
Default value:	**10**
Minimum:	1.0
Maximum:	99.0

Operands:
in | out
Must specify whether to zoom in and make larger or out and make smaller.

Command Index

1

1ddataset 29
1dplot 31
1drescale 32
1dtable 32

2

2ddataset 32
2dplot 33
2drescale 35

A

addfromhold 35
adjustangle 35
adjustdihedral 36
adjustdistance 37
alias 38
angle 38
appendribbons 38
application 39
atom 39
atomname 40
attach 40
autosetup 41

B

beginundoblock 42
bmincomfile 42
bond 42
bondorder 42
bondtonew 43
buildoptions 43

C

calcenergy 44
canonicalname 44
cascadepanels 45
cellsmarkerdump 45
cellsmarkersettings 45
centeratom 49
centerbond 49
centercoordinates 50
centroid 50
centroidatom 50
chainname 51
changedirectory 51
chiralatom 52
clip 52
clusteratom 53
clusterheavyatoms 53
clusterset 53
clustertorsion 54
coloratom 54
colorscheme 55
compareatom 55
compareset 56
confelim 56
confelimstart 57
confelimwrite 57
confgenltsearch 58
confgenmini 60
confgenpotential 61
confgenreadpotential 63
confgenstart 63
confgenwrite 64
confsearch 64
connect 66
connectfuseatom 66
constrainedangle 66
constrainedatom 67
constraineddist 68
constrainedset 69
constrainedtorsion 69
contactcriteria 70
contactset1 71
contactset2 72
createpropsubset 72
createsubset 73

D

defaultfc 73
delete 73
dihedral 74
dihedraldrive 75
displayatom 75
displayonlyatom 76
displayonlypose 76
displayopt 76
displaypose 80

distance 80
distancecheck 81
dynamics 81

E

ejob 83
elementlabels 84
embrace 84
embracecsearch 85
endundoblock 87
energykill 87
energymonitor 87
energysleep 88
energystart 88
energystop 89
energytask 89
energyupdate 89
energyupdatejobstatus 90
energywake 90
enhance3d 91
entrycombine 91
entrycopyprop 92
entrydelete 92
entrydisassociate 93
entrydragselection 93
entryduplicate 94
entryexport 94
entryexportspreadsheet 97
entryextendselection 98
entryextendwsinclude 98
entryimport 99
entryimportspreadsheet 101
entryimportvibration 102
entryinvertselection 102
entryrename 103
entryselect 103
entryselectall 103
entryselectonly 104
entryselectonlyrow 104
entryselectrandom 104
entryselectrow 105
entrysetprop 105
entrysettitle 106
entryshowall 107
entrytable 107
entryunselect 107
entryunselectall 108
entryunselectrow 108
entrywscreate 108
entrywsexclude 109
entrywsexcludenotfixed 109
entrywsinclude 109
entrywsincludelock 110
entrywsincludeonly 110
eplayergoto 110
eplayergotofirst 111
eplayergotolast 111
eplayersettings 111
eplayerstepahead 113
eplayerstepback 113
errorcheck 114
excludedvolumesmarkersettings 114
extenddisplaytopose 116

F

fileread 116
filewrite 118
filter 120
find 120
fit 123
forcefield 123
forcefieldbend 123
forcefieldele 124
forcefieldgbsolv 125
forcefieldimproper 125
forcefieldsasolv 126
forcefieldstretch 127
forcefieldtorsion 128
forcefieldvdw 128
forcefieldview 129
forcefieldwilson 129
formalcharge 130
fragment 131
fragmenttype 131
frozenatom 131
frozenset 132
fuse 132

G

geometrycleanup 132
glideactivesiteres 133
glideactivesiteset 133
glidecalcboxfromligand 133
glideconstraintregion 134
glideconstraintregionactivecell 134
glideconstraintregiongrow 134
glideconstraintregioninvisible 135
glideconstraintregionshrink 135
glideconstraintregionvisible 136

glidedisplayreceptor 136
glidedockregionnumatoms 136
glidedockregionselect 137
glidedockregionunselect 137
glidegridhydrophobic 137
glideligand 138
glideligandfile 140
glideligandgrid 141
glidelocatehydrophobiccells 142
glideoutput 142
glidereceptorligand 143
glidereferenceligand 143
glidescoring 144
glidesettings 146
glidesite 148
grow 150
growbond 150
growbond2 150
growdirection 151
growname 151

H

hbondcriteria 152
hbondset1 152
hbondset2 153
helpauto 153
helpcategory 154
helpsearch 154
helptopic 155
hideentries 156
hidepanel 156
hidepanels 156
hideproperty 156
historyvisible 157
hold 164
hydrogenapply 164
hydrogentreat 165

I

impactbufferedatom 165
impactbufferedset 165
impactconstraints 166
impactcontinuumsolvent 167
impactdynamics 167
impactfastmultipole 170
impactfrozenatom 170
impactfrozenset 171
impacthybridmc 171
impactjob 172
impactmdparams 173
impactminimization 174
impactperiodicboundary 176
impactpotential 177
impactstart 179
impacttask 179
impacttruncation 179
impactwrite 180
invert 180
invertset 181

J

jaguarassignatomnames 181
jaguarimportgeometry 181
jaguarinputfilesjob 183
jaguarjob 183
jaguarselectextendtablerow 184
jaguarselectonlytablerow 184
jobcleanup 185
jobsettings 185

L

labelatom 187
labelclear 191
labelformat 192
labelupdate 192
liaisonanalysis 192
liaisonparameters 195
liaisonselectlig 198
liaisonsettings 198
liaisonsystem 199
ligandbond 200
lightambient 201
lightdiffuse 201
lightposition 201
lightspecular 202
ligprep 202
ligprepstart 204
localatoms 204
localbitset 205
localcenter 205
logp 205

M

macrodefine 206
macrorun 206
materialmolecular 207
mcsd 208
minienergy 208
minta 210
monitorangle 211
monitordistance 211
monitorhbond 212
monitorsetsurf 213
monitorsettings 213
monitorsurf 214
monitortorsion 214
mouse 214
move 215
multiplemini 215
mutate 216

N

nextpose 216

O

optimizefog 217

P

partialcharge 217
pause 218
pausecommands 218
phaseaddcustomfeature 218
phaseaddligands 219
phaseaddligandsfromrun 219
phasebuildqsar 220
phasechooseactiveset 221
phasecleanupstructures 221
phaseclearcentroidatoms 221
phaseconfgen 221
phasecreateexcludedvolume 223
phasedbcleanupstructures 224
phasedbclose 225
phasedbconfgen 225
phasedbcreatesites 227
phasedbcreatesubset 227
phasedbdelete 227
phasedbdeletesubset 228
phasedbgenerateconformers 228
phasedbimport 228
phasedbimportfile 229
phasedbimportrun 229
phasedbnewfromfile 229
phasedbopen 230
phasedbrenamesubset 230
phasedbresetall 230
phasedbselectsubset 231
phasedbsetstorage 231
phasedbstepforward 232
phasedbstepgoto 232
phasedeletecustomfeature 232
phasedeleteexcludedvolumes 233
phasedeletehypothesis 233
phasedeletehypothesisfrombuilder 233
phasedeleteselectedligands 234
phasedisplayproperty 234
phaseexcludetablerow 234
phaseexportfeature 234
phaseexporthypothesis 235
phaseexporthypothesisfrombuilder 235
phaseexportselectedalignments 236
phaseexportselectedligands 236
phasefindmatches 236
phasefindpharmacophores 238
phasegenerateconformers 239
phasehookimport 239
phasehypothesiscreateexcludedvolume 239
phasehypothesisdeleteexcludedvolumes 239
phasehypothesisselectevrow 240
phasehypothesisselectextendevrow 240
phasehypothesisselectonlyevrow 240
phasehypothesissetexcludedvolumes 241
phasehypothesistoggleinclude 241
phasehypothesisunselectevrow 241
phaseimportfeature 242
phaseimporthypothesis 242
phaseincludeextendtablerow 243
phaseincludeonlytablerow 243
phaseincludetablerow 243
phaseinverttableselection 244
phaseligprep 244
phaselocatesites 245
phasemarkerdump 245
phasemarkersettings 245
phasemarkfeature 256

phaseoptions 256
phasepatterndelete 256
phasepatternedit 257
phasepatternmovedown 258
phasepatternmoveup 258
phasepatternnew 259
phasepatternsetoptions 260
phasepreview 261
phaseqsarsearch 261
phaserandomtraining 261
phaserandomtrainingoptions 261
phaserefreshfrequencytable 262
phaseremoveligands 262
phaserenameligand 262
phaserescorehypotheses 263
phaserescoreweighting 263
phaseresetfeatures 264
phaseruncreate 264
phaserundelete 264
phaserunopen 264
phaserunrename 265
phaserunsaveas 265
phaserunsetseed 265
phaserunsetstereo 266
phasescorehypotheses 266
phascoreselecthypothesis 268
phascoreweighting 269
phaseselectalltablerows 270
phaseselectevrow 270
phaseselectextendevrow 270
phaseselectextendtablerow 270
phaseselectonlyevrow 271
phaseselectonlytablerow 271
phaseselecttablerow 272
phasesetactivity 272
phasesetactivityproperty 272
phasesetexcludedvolumes 273
phasesetfrequency 273
phasesettrainingrows 274
phasesiteoptions 274
phasesorttable 275
phasestepforward 276
phasestepgoto 276
phasetoggleactivetablerow 276
phasetogglecentroidatom 277
phasetoggletrainingrow 277
phaseundisplayproperty 277
phaseunmarkfeature 278
phaseunselectevrow 278
phaseunselecttablerow 278
picksize 279
place 279
plotxyarrangecolumn 279
plotxyarrangerow 280
plotxyarrangetiled 280
plotxyaxis 280
plotxyaxisautorange 281
plotxyaxisdelete 282
plotxyaxisdisplay 282
plotxyaxisrename 283
plotxycaption 283
plotxycaptionposition 284
plotxycopy 284
plotxydelete 284
plotxydisplay 285
plotxydisplaycaption 285
plotxydisplayincluded 285
plotxydisplaylegend 286
plotxydisplayname 286
plotxydisplayonly 286
plotxydisplaypointlabels 287
plotxydisplayselected 287
plotxyhidesidebar 287
plotxyhidetoolbar 288
plotxylabel 288
plotxymove 289
plotxynew 289
plotxypan 289
plotxypanplot 290
plotxyrename 290
plotxyresetview 290
plotxysaveimage 291
plotxyselect 291
plotxyseries 291
plotxyseriesdelete 294
plotxyseriesdisplay 294
plotxyseriesrename 294
plotxyseriesselect 295
plotxyseriesselectadd 295
plotxyshowsidebar 296
plotxyshowtoolbar 296
plotxytoggledisplay 296
plotxyundisplay 297
plotxyunlabel 297
plotxyunselect 297
plotxyupdate 298
plotxyzoom 298
plotxyzoompan 298
posereceptor 299
potential 299
pprep 301
pprepreceptorligand 302

pprepwrite 302
prefer 302
previouspose 308
print 308
printoptions 308
projectclose 310
projectcopy 310
projectdelete 311
projectmerge 311
projectnew 312
projectopen 313
projectrename 313
projectrevertopen 313
projectrevertsnapshot 313
projectsaveas 314
projectstoresnapshot 314
projectsynchronize 314
projecttablefind 315
projectupdatecoordinates 315
propertycalculate 316
propertycalculatemolformula 317
propertycalculatepickpka 317
propertycalculatesubstructs 318
propertyclearvalue 319
propertycreate 320
propertydelete 321
propertygeneratecontacts 321
propertygeneratehbond 321
propertymeasurementsetting 322
propertymove 322
propertyprecision 323
propertyrename 323
propertyshowall 323
propertysuperimposesetting 324
pspalignaddanchor 324
pspalignndeleteanchor 324
pspaligninsertgaps 325
pspalignlockgaps 325
pspalignmoveleft 326
pspalignmoveleftblock 326
pspalignmoveright 327
pspalignstructures 327
pspalignunlockgaps 328
pspbstogglehetatom 328
pspbuildbackbone 328
pspbuildstructure 329
pspexcludetable1row 329
pspexcludetablerow 329
pspexportalignment 330
pspfindfamily 330
pspfindhomologs 330
pspfoldrecognitionoptions 330
pspfoldrecognitionsearch 331
pspimportalignment 331
pspimporthomolog 332
pspimportssp 332
pspincludetable1row 332
pspincludetablerow 333
pspminimizationresidues 333
pspoptimizealignment 333
psprefinebackbone 334
psprefinestructure 334
psprsdefaulthelixloops 334
psprshelixlooptogglerefine 334
psprshelixoptions 335
psprsloopoptions 336
psprunalign 337
pspruncreate 338
psprundelete 338
psprunopen 338
psprunrename 338
psprunsaveas 339
pspsecstructprediction 339
pspselectextendtable1row 339
pspselectextendtablerow 340
pspselecthelixlooprow 340
pspselectonlytable1row 341
pspselectonlytablerow 341
pspselectrscontext 342
pspselectrsrefinement 342
pspselecttable1row 342
pspselecttablerow 343
pspsequenceaddfile 343
pspsequenceaddworkspace 343
pspsequencecrop 344
pspsequenceselect 344
pspsequenceviewerexport 344
pspsethelixloopresidues 345
pspsidechainresidues 345
pspsortbbtable 345
pspsortfindhomologstable 346
pspsortfoldtable 346
pspsortrbtable 347
pspsortrbtable1 347
pspsortrstable 347
pspsspdelete 348
pspsspexport 348
pspssprevert 348
pspsspset 349
pspstepforward 349
pspstepgoto 350
pspstructureaddentry 350

psptemplatesetregion 350
psptmplsecstructprediction 351
pspunselecttable1row 351
pspunselecttablerow 352
pspupdatescores 352
pythoneval 353
pythonimport 353
pythonrun 354

Q

qikprop 354
qsarmarkerdump 355
qsarmarkersettings 355
qsiteion 357
qsiteresidue 358
qsiteset 358
quit 360

R

readposefile 361
readpotential 361
refinestart 361
refinewrite 362
rename 362
repall 362
repatom 365
repatombonds 366
repbond 366
repdefault 366
replacefromhold 367
repquick 367
resetcsearch 368
residuename 368
residuenumber 369
residuerenumber 369
restorepanels 370
retype 370
ribbon 370
ribbondump 379
ringclosure 379
rotate 380

S

saveimage 381
saveimageheight 382
saveimagewidth 382
savelayout 382
scriptlogfile 383
scriptrun 383
searchdbconfgen 384
set 386
setread 386
setwrite 386
showhwstereosetup 387
showpanel 387
showpanels 387
sitemap 388
sitemapbox 389
sitemapset 389
sitemapstart 390
sitemapwrite 390
specifiedname 390
spotcenter 391
strikebuildqsar 391
strikedeletemodel 393
strikeexportmodel 393
strikeextendselectdescriptor 393
strikeimportmodel 394
strikepredict 394
strikeselectdescriptor 394
strikeselectmodel 395
strikesimilarity 395
striketoggleselectdescriptor 395
structalignatoms 396
structalignstart 396
substructure 396
substructurefileread 397
substructurefilewrite 397
substructureshell 398
superimpose 399
superimposeatom 399
superimposeset 400
surfacedelete 400
surfacedisplay 401
surfaceduplicate 401
surfaceextended 402
surfaceextendedradiuscontext 403
surfaceextendedradiusset 403
surfacemolecular 403
surfacemolecularcontext 404
surfacemolecularset 405
surfacerename 405
surfacescheme 406
surfacesetcomment 407
surfacesetisovalue 407
surfacesetviewasl 408
surfacestyle 409
surfacetransparency 409
surfacevdw 410

surfacevdwcontext 411
surfacevdwset 411
surfaceviewaslset 412
symmetrizeworkspace 412
system 413

T

tableresizecolumn 413
tablesort 413
tablesortall 414
tablesortfields 414
tableunselectnonsubset 415
tile 415
toolbarmain 415
torsioncheck 415
torsiongroup 416
transform 417
translate 418

U

undisplayatom 418
undisplaypose 419
undo 419
unhookimport 419
uniquename 420
uniquepdb 420
updateribbons 420

V

varymolecule 421
varytorsion 421
viewmatrix 422
viewposecontacts 422
viewposehbonds 423
viewreset 423
viewrestore 423
viewsave 423
viewvolume 424
visimport 425
volumedelete 425
volumeisosurface 426
volumerename 426

W

workspaceselectionadd 427
workspaceselectionclear 428
workspaceselectioninvert 428
workspaceselectionreplace 428
workspaceselectionsubtract 428
writeangle 429
writecontact 430
writedihedral 431
writedistance 432
writehbond 432
writeposefile 433

X

xcluster 434
xclusterstart 435
xclusterwrite 435

Z

zoom 435

Table of Contents

1 The Maestro Command Language 1

2 Conventions Used in this Manual 3

3 The Atom Specification Language 5

3.1 Why an Atom Specification Language? 5
3.2 The ASL Hierarchy 5
3.3 Atom Specification 6
3.4 Operators 13
3.4.1 The Boolean `and` Operator 13
3.4.2 The Boolean `or` Operator 13
3.4.3 The Boolean `not` Operator 14
3.4.4 The `fillres` and `fillmol` Operators 14
3.4.5 The `within` and `beyond` Operators 14
3.4.6 The `withinbonds` Operator 15
3.4.7 The `beyondbonds` Operator 15
3.5 Operator Priority 15
3.6 Implicit Operators 16
3.7 Creating New Sets from Existing Ones 16
3.8 Special Specifications 17
3.9 Matching PDB Atom Names 17
3.10 Miscellaneous 18
3.10.1 Atoms not yet present in a structure 18
3.10.2 Aliasing 18
3.11 Useful Hints when using ASL with the Project Facility 19
3.12 ASL Examples 20

4 The Entry Specification Language 25

4.1 Why an Entry Selection Language? 25
4.2 Entry Properties 25
4.3 Logical Operators 26
4.4 Entry Property Comparisons 26
4.5 Examples 27

5 Commands 29
1ddataset 29
1dplot 31
1drescale 32
1dtable 32
2ddataset 32
2dplot 33
2drescale 34
addfromhold 35
adjustangle 35
adjustdihedral 36
adjustdistance 37
alias 38
angle 38
appendribbons 38
application 39
atom 39
atomname 40
attach 40
autosetup 41
beginundoblock 41
bmincomfile 42
bond 42
bondorder 42
bondtonew 43
buildoptions 43
calcenergy 44
canonicalname 44
cascadepanels 45
cd 45
cellsmarkerdump 45
cellsmarkersettings 45
centeratom 49
centerbond 49
centercoordinates 50
centroid 50
centroidatom 50
chainname 51
changedirectory 51
chdir 52
chiralatom 52
clip 52
clusteratom 53
clusterheavyatoms 53
clusterset 53

clustertorsion 54
coloratom 54
colorscheme 55
compareatom 55
compareset 56
confelim 56
confelimstart 57
confelimwrite 57
confgenltsearch 58
confgenmini 60
confgenpotential 61
confgenreadpotential 63
confgenstart 63
confgenwrite 63
confsearch 64
connect 66
connectfuseatom 66
constrainedangle 66
constrainedatom 67
constraineddist 68
constrainedset 69
constrainedtorsion 69
contactcriteria 70
contactset1 71
contactset2 72
createpropsubset 72
createsubset 72
defaultfc 73
delete 73
deleteproperty 74
dihedral 74
dihedraldrive 75
displayatom 75
displayonlyatom 76
displayonlypose 76
displayopt 76
displaypose 80
distance 80
distancecheck 80
dynamics 81
ecalc 82
ejob 82
elementlabels 83
embrace 84
embracecsearch 85

endundoblock . . . 87
energykill . . . 87
energymonitor . . . 87
energysleep . . . 88
energystart . . . 88
energystop . . . 88
energytask . . . 89
energyupdate . . . 89
energyupdatejobstatus . . . 90
energywake . . . 90
enhance3d . . . 90
entrycombine . . . 91
entrycopyprop . . . 91
entrydelete . . . 92
entrydisassociate . . . 93
entrydragselection . . . 93
entryduplicate . . . 94
entryexport . . . 94
entryexportspreadsheet . . . 97
entryextendselection . . . 98
entryextendwsinclude . . . 98
entryimport . . . 98
entryimportspreadsheet . . . 101
entryimportvibration . . . 102
entryinvertselection . . . 102
entryrename . . . 102
entryselect . . . 103
entryselectall . . . 103
entryselectonly . . . 104
entryselectonlyrow . . . 104
entryselectrandom . . . 104
entryselectrow . . . 105
entrysetprop . . . 105
entrysettitle . . . 106
entryshowall . . . 106
entrytable . . . 107
entryunselect . . . 107
entryunselectall . . . 107
entryunselectrow . . . 108
entrywscreate . . . 108
entrywsexclude . . . 109
entrywsexcludenotfixed . . . 109
entrywsinclude . . . 109
entrywsincludelock . . . 109
entrywsincludeonly . . . 110

eplayergoto 110
eplayergotofirst 111
eplayergotolast 111
eplayersettings 111
eplayerstepahead 113
eplayerstepback 113
errorcheck 113
excludedvolumesmarkersettings 114
extenddisplaytopose 115
fileread 116
filewrite 118
filter 120
find 120
fit 123
forcefield 123
forcefieldbend 123
forcefieldele 124
forcefieldgbsolv 125
forcefieldimproper 125
forcefieldsasolv 126
forcefieldstretch 127
forcefieldtorsion 127
forcefieldvdw 128
forcefieldview 129
forcefieldwilson 129
formalcharge 130
fragment 130
fragmenttype 131
frozenatom 131
frozenset 132
fuse 132
geometrycleanup 132
glideactivesiteres 133
glideactivesiteset 133
glidecalcboxfromligand 133
glideconstraintregion 134
glideconstraintregionactivecell 134
glideconstraintregiongrow 134
glideconstraintregioninvisible 135
glideconstraintregionshrink 135
glideconstraintregionvisible 135
glidedisplayreceptor 136
glidedockregionnumatoms 136
glidedockregionselect 136
glidedockregionunselect 137

glidegridhydrophobic 137
glideligand 138
glideligandfile 140
glideligandgrid 141
glidelocatehydrophobiccells 142
glideoutput 142
gliderecepto rligand 143
glidereferenceligand 143
glidescoring 144
glidesettings 146
glidesite 148
grow 150
growbond 150
growbond2 150
growdirection 151
growname 151
happly 151
hbondcriteria 152
hbondset1 152
hbondset2 153
help 153
helpauto 153
helpcategory 154
helpsearch 154
helptopic 155
hideentries 155
hidepanel 156
hidepanels 156
hideproperty 156
historyvisible 157
hold 164
htreat 164
hydrogenapply 164
hydrogentreat 165
impactbufferedatom 165
impactbufferedset 165
impactconstraints 166
impactcontinuumsolvent 167
impactdynamics 167
impactfastmultipole 169
impactfrozenatom 170
impactfrozenset 170
impacthybridmc 171
impactjob 171
impactmdparams 172

impactminimization 174
impactperiodicboundary 176
impactpotential 177
impactstart 179
impacttask 179
impacttruncation 179
impactwrite 180
invert 180
invertset 180
jaguarassignatomnames 181
jaguarimportgeometry 181
jaguarinputfilesjob 182
jaguarjob 183
jaguarselectextendtablerow 183
jaguarselectonlytablerow 184
jobcleanup 184
jobsettings 185
kill 187
labelatom 187
labelclear 191
labelformat 191
labelupdate 192
liaisonanalysis 192
liaisonparameters 195
liaisonselectlig 198
liaisonsettings 198
liaisonsystem 199
ligandbond 200
lightambient 201
lightdiffuse 201
lightposition 201
lightspecular 202
ligprep 202
ligprepstart 204
localatoms 204
localbitset 205
localcenter 205
logfile 205
logp 205
macrodefine 206
macrorun 206
materialmolecular 207
mcsd 207
mini 208
minienergy 208

minta . . . 209
monitor . . . 211
monitorangle . . . 211
monitordistance . . . 211
monitorhbond . . . 212
monitorsetsurf . . . 213
monitorsettings . . . 213
monitorsurf . . . 213
monitortorsion . . . 214
mouse . . . 214
move . . . 215
multiplemini . . . 215
mutate . . . 216
nextpose . . . 216
optimizefog . . . 217
partialcharge . . . 217
pause . . . 217
pausecommands . . . 218
phaseaddcustomfeature . . . 218
phaseaddligands . . . 219
phaseaddligandsfromrun . . . 219
phasebuildqsar . . . 220
phasechooseactiveset . . . 220
phasecleanupstructures . . . 221
phaseclearcentroidatoms . . . 221
phaseconfgen . . . 221
phasecreateexcludedvolume . . . 223
phasedbcleanupstructures . . . 223
phasedbclose . . . 225
phasedbconfgen . . . 225
phasedbcreatesites . . . 227
phasedbcreatesubset . . . 227
phasedbdelete . . . 227
phasedbdeletesubset . . . 228
phasedbgenerateconformers . . . 228
phasedbimport . . . 228
phasedbimportfile . . . 228
phasedbimportrun . . . 229
phasedbnewfromfile . . . 229
phasedbopen . . . 230
phasedbrenamesubset . . . 230
phasedbresetall . . . 230
phasedbselectsubset . . . 231
phasedbsetstorage . . . 231
phasedbstepforward . . . 232

phasedbstepgoto 232
phasedeletecustomfeature 232
phasedeleteexcludedvolumes 233
phasedeletehypothesis 233
phasedeletehypothesisfrombuilder 233
phasedeleteselectedligands 233
phasedisplayproperty 234
phaseexcludetablerow 234
phaseexportfeature 234
phaseexporthypothesis 235
phaseexporthypothesisfrombuilder 235
phaseexportselectedalignments 235
phaseexportselectedligands 236
phasefindmatches 236
phasefindpharmacophores 238
phasegenerateconformers 238
phasehookimport 239
phasehypothesiscreateexcludedvolume 239
phasehypothesisdeleteexcludedvolumes 239
phasehypothesisselectevrow 239
phasehypothesisselectextendevrow 240
phasehypothesisselectonlyevrow 240
phasehypothesissetexcludedvolumes 240
phasehypothesistoggleinclude 241
phasehypothesisunselectevrow 241
phaseimportfeature 242
phaseimporthypothesis 242
phaseincludeextendtablerow 243
phaseincludeonlytablerow 243
phaseincludetablerow 243
phaseinverttableselection 244
phaseligprep 244
phaselocatesites 245
phasemarkerdump 245
phasemarkersettings 245
phasemarkfeature 255
phaseoptions 256
phasepatterndelete 256
phasepatternedit 257
phasepatternmovedown 258
phasepatternmoveup 258
phasepatternnew 259
phasepatternsetoptions 260
phasepreview 261
phaseqsarsearch 261

phaserandomtraining 261
phaserandomtrainingoptions 261
phaserefreshfrequencytable 262
phaseremoveligands 262
phaserenameligand 262
phaserescorehypotheses 263
phaserescoreweighting 263
phaseresetfeatures 264
phaseruncreate 264
phaserundelete 264
phaserunopen 264
phaserunrename 265
phaserunsaveas 265
phaserunsetseed 265
phaserunsetstereo 266
phasescorehypotheses 266
phasescoreselecthypothesis 268
phasescoreweighting 269
phaseselectalltablerows 269
phaseselectevrow 270
phaseselectextendevrow 270
phaseselectextendtablerow 270
phaseselectonlyevrow 271
phaseselectonlytablerow 271
phaseselecttablerow 271
phasesetactivity 272
phasesetactivityproperty 272
phasesetexcludedvolumes 273
phasesetfrequency 273
phasesettrainingrows 274
phasesiteoptions 274
phasesorttable 275
phasestepforward 275
phasestepgoto 276
phasetoggleactivetablerow 276
phasetogglecentroidatom 276
phasetoggletrainingrow 277
phaseundisplayproperty 277
phaseunmarkfeature 277
phaseunselectevrow 278
phaseunselecttablerow 278
picksize 278
place 279
plotxyarrangecolumn 279
plotxyarrangerow 279

plotxyarrangetiled 280
plotxyaxis 280
plotxyaxisautorange 281
plotxyaxisdelete 281
plotxyaxisdisplay 282
plotxyaxisrename 283
plotxycaption 283
plotxycaptionposition 284
plotxycopy 284
plotxydelete 284
plotxydisplay 285
plotxydisplaycaption 285
plotxydisplayincluded 285
plotxydisplaylegend 286
plotxydisplayname 286
plotxydisplayonly 286
plotxydisplaypointlabels 286
plotxydisplayselected 287
plotxyhidesidebar 287
plotxyhidetoolbar 287
plotxylabel 288
plotxymove 289
plotxynew 289
plotxypan 289
plotxypanplot 290
plotxyrename 290
plotxyresetview 290
plotxysaveimage 290
plotxyselect 291
plotxyseries 291
plotxyseriesdelete 293
plotxyseriesdisplay 294
plotxyseriesrename 294
plotxyseriesselect 295
plotxyseriesselectadd 295
plotxyshowsidebar 296
plotxyshowtoolbar 296
plotxytoggledisplay 296
plotxyundisplay 296
plotxyunlabel 297
plotxyunselect 297
plotxyupdate 298
plotxyzoom 298
plotxyzoompan 298
posereceptor 299

potential 299
pprep 301
pprepreceptorligand 302
pprepwrite 302
prefer 302
previouspose 308
print 308
printoptions 308
projectclose 310
projectcopy 310
projectdelete 311
projectmerge 311
projectnew 312
projectopen 312
projectrename 313
projectrevertopen 313
projectrevertsnapshot 313
projectsaveas 314
projectstoresnapshot 314
projectsynchronize 314
projecttablefind 314
projectupdatecoordinates 315
propertycalculate 315
propertycalculatemolformula 316
propertycalculatepickpka 317
propertycalculatesubstructs 318
propertyclearvalue 319
propertycreate 319
propertydelete 321
propertygeneratecontacts 321
propertygeneratehbond 321
propertymeasurementsetting 322
propertymove 322
propertyprecision 322
propertyrename 323
propertyshowall 323
propertysuperimposesetting 323
pspalignaddanchor 324
pspaligndeleteanchor 324
pspaligninsertgaps 325
pspalignlockgaps 325
pspalignmoveleft 325
pspalignmoveleftblock 326
pspalignmoveright 327
pspalignstructures 327

pspalignunlockgaps 328
pspbstogglehetatom 328
pspbuildbackbone 328
pspbuildstructure 328
pspexcludetable1row 329
pspexcludetablerow 329
pspexportalignment 330
pspfindfamily 330
pspfindhomologs 330
pspfoldrecognitionoptions 330
pspfoldrecognitionsearch 331
pspimportalignment 331
pspimporthomolog 331
pspimportssp 332
pspincludetable1row 332
pspincludetablerow 333
pspminimizationresidues 333
pspoptimizealignment 333
psprefinebackbone 334
psprefinestructure 334
psprsdefaulthelixloops 334
psprshelixlooptogglerefine 334
psprshelixoptions 335
psprsloopoptions 336
psprunalign 337
pspruncreate 337
psprundelete 338
psprunopen 338
psprunrename 338
psprunsaveas 339
pspsecstructprediction 339
pspselectextendtable1row 339
pspselectextendtablerow 340
pspselecthelixlooprow 340
pspselectonlytable1row 341
pspselectonlytablerow 341
pspselectrscontext 341
pspselectrsrefinement 342
pspselecttable1row 342
pspselecttablerow 342
pspsequenceaddfile 343
pspsequenceaddworkspace 343
pspsequencecrop 343
pspsequenceselect 344
pspsequenceviewerexport 344

pspsethelixloopresidues 345
pspsidechainresidues 345
pspsortbbtable 345
pspsortfindhomologstable 346
pspsortfoldtable 346
pspsortrbtable 346
pspsortrbtable1 347
pspsortrstable 347
pspsspdelete 347
pspsspexport 348
pspssprevert 348
pspsspset 349
pspstepforward 349
pspstepgoto 350
pspstructureaddentry 350
psptemplatesetregion 350
psptmplsecstructprediction 351
pspunselecttable1row 351
pspunselecttablerow 352
pspupdatescores 352
pyeval 352
pyimp 353
pyrun 353
pythoneval 353
pythonimport 353
pythonrun 354
qikprop 354
qsarmarkerdump 354
qsarmarkersettings 355
qsiteion 357
qsiteresidue 357
qsiteset 358
quit 360
read 360
readposefile 361
readpotential 361
refinestart 361
refinewrite 361
rename 362
renameproperty 362
repall 362
repatom 365
repatombonds 365
repbond 366
repdefault 366

replacefromhold 367
repquick 367
resetcsearch 368
residuename 368
residuenumber 369
residuerenumber 369
restorepanels 370
retype 370
ribbon 370
ribbondump 379
ringclosure 379
rotate 380
run 381
saveimage 381
saveimageheight 382
saveimagewidth 382
savelayout 382
scriptlogfile 383
scriptrun 383
searchdbconfgen 384
set 386
setread 386
setwrite 386
showhwstereosetup 387
showpanel 387
showpanels 387
sitemap 388
sitemapbox 389
sitemapset 389
sitemapstart 390
sitemapwrite 390
sleep 390
specifiedname 390
spotcenter 391
stop 391
strikebuildqsar 391
strikedeletemodel 392
strikeexportmodel 393
strikeextendselectdescriptor 393
strikeimportmodel 393
strikepredict 394
strikeselectdescriptor 394
strikeselectmodel 394
strikesimilarity 395
striketoggleselectdescriptor 395

structalignatoms 396
structalignstart 396
substructure 396
substructurefile 397
substructurefileread 397
substructurefilewrite 397
substructureshell 398
superimpose 399
superimposeatom 399
superimposeset 400
surfacedelete 400
surfacedisplay 401
surfaceduplicate 401
surfaceextended 402
surfaceextendedradiuscontext 403
surfaceextendedradiusset 403
surfacemolecular 403
surfacemolecularcontext 404
surfacemolecularset 405
surfacerename 405
surfacescheme 406
surfacesetcomment 407
surfacesetisovalue 407
surfacesetviewasl 408
surfacestyle 409
surfacetransparency 409
surfacevdw 410
surfacevdwcontext 411
surfacevdwset 411
surfaceviewaslset 412
symmetrizeworkspace 412
system 413
tableresizecolumn 413
tablesort 413
tablesortall 414
tablesortfields 414
tableunselectnonsubset 414
tile 415
toolbarmain 415
torsioncheck 415
torsiongroup 416
transform 416
translate 418
undisplayatom 418
undisplaypose 419

undo . . . 419
unhookimport . . . 419
uniquename . . . 419
uniquepdb . . . 420
update . . . 420
updateribbons . . . 420
varymolecule . . . 420
varytorsion . . . 421
viewmatrix . . . 422
viewposecontacts . . . 422
viewposehbonds . . . 423
viewreset . . . 423
viewrestore . . . 423
viewsave . . . 423
viewvolume . . . 424
visimport . . . 424
volumedelete . . . 425
volumeisosurface . . . 426
volumerename . . . 426
wake . . . 427
workspaceselectionadd . . . 427
workspaceselectionclear . . . 427
workspaceselectioninvert . . . 428
workspaceselectionreplace . . . 428
workspaceselectionsubtract . . . 428
write . . . 429
writeangle . . . 429
writecontact . . . 430
writedihedral . . . 431
writedistance . . . 431
writehbond . . . 432
writeposefile . . . 433
xcluster . . . 434
xclusterstart . . . 435
xclusterwrite . . . 435
zoom . . . 435

Command Index . . . 437